radisys

A6K-RSM-J
SHELF MANAGER

SOFTWARE TECHNICAL PRODUCT SPECIFICATION

January 2012 **007-03370-0003**

Revision history

Version	Date	Description
-0000	September 2010	First edition.
-0001	May 2011	Second edition. Updated values for voltage and temperature threshold sensors in Table 9 on page 31. Revised event output strings in Table 92 and Table 170. Removed 0030 and 0036 event codes from Table 85 on page 226. Noted in *Fantray Control Mode* on page 119 that fan tray local control mode is not supported. Added Setting/Getting the Active Network Direction procedures on page 159. Added *Setting Ethernet Bonding* on page 164. Added `POWERON_IGNORE_CRITICAL_TEMP_SHELF` parameter for configuring the cooling policy. Added Filter Run Time shelf sensor. Revised the FRU Update Utility chapter to include information about FRU data recovery and command options for the `fru_update` utility.
-0002	September 2011	Third edition. New Radisys document branding; fixed broken links; corrected Table 125 on page 249 and Table 138 on page 258 to remove the open ejector request event.
-0003	January 2012	Fourth edition. See *What's New in This Manual* on page 15 for a description of the changes in this edition.

© 2010-2012 by Radisys Corporation. All rights reserved.

Radisys and Procelerant are registered trademarks of Radisys Corporation. AdvancedTCA, ATCA, and PICMG are registered trademarks of PCI Industrial Computer Manufacturers Group. Wind River is a registered trademark of Wind River Systems Inc. Red Hat and Enterprise Linux are registered trademarks of Red Hat Inc. Procomm Plus and Symantec are registered trademarks of Symantec Corporation. Intel is a registered trademark of Intel Corporation. Linux is a registered trademark of Linus Torvalds.

All other trademarks, registered trademarks, service marks, and trade names are the property of their respective owners.

Table of Contents

1.0 Document Organization 14
1.1 Document Organization 14
1.2 What's New in This Manual 15
1.3 Glossary of Terms Used in This Document 16

2.0 Introduction 18
2.1 Overview 18
2.2 AdvancedMC* Support 18
2.3 Third-party Chassis Integration 18
2.4 Specification Conformance 18
2.5 Related Documents 19

3.0 System Level Specifications 21
3.1 U-Boot* 21
3.2 Operating System 21
3.3 File System Organization 21
3.3.1 Flash Storage 22
3.4 Random Access Memory 23
3.5 Configuration Files 23
3.6 Factory Reset 23
3.7 Application Hosting 23
3.7.1 Startup and Shutdown Scripts 23
3.7.2 Available System Resources 24
3.8 System Management Interfaces 24
3.9 Ethernet Interfaces 26
3.10 IPMB 26
3.11 Telco Alarms 26

4.0 Front Panel LEDs 27
4.1 LED Types and States 27
4.1.1 Power Good LED 27
4.1.2 Hot Swap LED 27
4.1.3 Active LED 27
4.1.4 Out of Service LED 28
4.2 Retrieving a Location's LED Properties 28
4.3 Retrieving Color Properties of LEDs 28
4.4 Retrieving State of LEDs 28
4.5 Using Lamptest Function 28
4.6 LED Boot Sequence 28

5.0 Sensors 30
5.1 Overview 30
5.2 Threshold-based Sensors 30
5.2.1 Threshold-based Sensors on RSM 30
5.3 Discrete Sensors 32
5.3.1 OEM Sensors 32
5.4 Sensor Event Description String 32
5.5 Sensor Information Details 33
5.5.1 SEL Entries 33
5.5.2 SNMP Traps 33
5.6 Sensor Targets 33

6.0 Health Events 34
6.1 Overview 34
6.2 Health Queries 34

6.3 Healthevents Queries 34
6.3.1 Healthevents Queries for Individual Sensors 35
6.3.2 Healthevents Queries for All Sensors on Location 35
6.3.3 No Active Events 36
6.3.4 Not Present or Non-IPMI Locations 36
6.4 Health Event Property Configuration 36

7.0 Alarms 37
7.1 Overview 37
7.2 Annunciators 37
7.3 Acknowledging Alarms 37

8.0 System Event Log 38
8.1 SEL Architecture on RSM 38
8.2 Retrieving SEL 38
8.3 SEL Display Format 39
8.3.1 Header 39
8.3.2 Text Translation 39
8.3.3 Raw Output 39
8.3.4 Configuring SEL Display Format 40
8.3.5 Displaying Unrecognized SEL Events 40
8.4 Retrieving SEL in Raw Format 41
8.5 Clearing SEL 41
8.6 SEL Configuration 41

9.0 Trap Generation and Platform Event Filtering 42
9.1 Trap Generation and Platform Event Filtering 42
9.2 Configuration 42
9.2.1 Event Filtering Method 42
9.2.2 PEF Filter 43
9.2.3 PEF Alert Policy 44
9.2.4 PEF Alert String 44
9.2.5 System GUID 45
9.3 Supported PEF Functionality 46
9.4 PET Trap 47

10.0 High Availability 49
10.1 Overview 49
10.2 Readiness State 49
10.2.1 Changing Peer RSM Readiness State 50
10.2.2 HA Redundancy Sensor 50
10.3 HA State 50
10.3.1 Presence State 51
10.3.2 HA State Sensor 51
10.3.3 In-service Request Sensor 52
10.3.4 Out-of-service Request Sensor 52
10.3.5 Redundancy Sensor 52
10.4 Health Score 52
10.4.1 Health Score Sensor 52
10.5 Data Synchronization 53
10.5.1 Time and Date Synchronization 54
10.5.2 User Scripts Synchronization 54
10.5.3 Data Synchronization Failure 55
10.5.4 Heterogeneous Synchronization 55
10.5.5 DataSync Status Sensor 55

10.6 Failover and Switchover 56
10.6.1 Switchover 56
10.6.2 Failover 58
10.6.3 Standby Reboot 58
10.6.4 HA Control Sensor 58
10.7 CMM Status Sensor 58

11.0 Re-enumeration 59
11.1 Overview 59
11.2 Re-enumeration Sensor 59
11.3 Event Regeneration 59
11.4 Cooling 59
11.5 Resolution of EKeys 60

12.0 Process Monitoring and Integrity 61
12.1 Overview 61
12.1.1 Process Existence Monitoring 61
12.1.2 Process Watchdog Monitoring 61
12.1.3 Process Integrity Monitoring 62
12.2 Processes Monitored 62
12.3 Process Monitoring Targets 62
12.4 Process Dependency 63
12.5 Peer Processes 63
12.6 Process Monitoring Dataitems 64
12.6.1 Examples 64
12.7 Process Monitoring RSM Events 64
12.8 Failure Scenarios and Event Processing 65
12.8.1 No action recovery 65
12.8.2 Successful restart recovery 66
12.8.3 Successful failover and restart recovery 66
12.8.4 Successful failover and reboot recovery 66
12.8.5 Failed failover and reboot recovery for a non-critical process 67
12.8.6 Failed failover and reboot recovery for a critical process 68
12.8.7 Excessive restarts and escalation is no action 68
12.8.8 Excessive restarts and successful failover/reboot escalation 69
12.8.9 Excessive restarts, failed failover/reboot escalation, non-critical process 70
12.8.10 Excessive restarts, failed failover/reboot escalation, critical process 70
12.8.11 Process administrative action 71
12.9 Configuration 71
12.9.1 Configuration Parameters 72

13.0 Security 76
13.1 Role-based Access Control 76
13.2 User Management 76
13.3 Security Sensor 77

14.0 Hardware Platform Interface 78
14.1 Overview 78
14.2 OpenHPI* 78
14.3 RSM Plug-in to OpenHPI* 78

15.0 Shelf Management & OAM API 79
15.1 Overview 79
15.2 Shelf Management and OAM API Client Library 79
15.3 ShM API Access Permissions 79

16.0 Command Line Interface 81
16.1 Overview 81

17.0 Simple Network Management Protocol 82
17.1 Net-SNMP* 82
17.2 Supported MIBs 82
17.2.1 Chassis Management Module MIB 82
17.2.2 OAM MIB 82
17.2.3 MIB II 82
17.3 Use of Sub-FRUs 83
17.4 Third-party Chassis Support 84
17.4.1 Fan Tray 84
17.4.2 Power Entry Module 84
17.4.3 Air Filter Tray 84
17.4.4 Shelf FRU 84
17.4.5 SAP 84
17.4.6 Alias Mappings 85
17.5 SNMP Agent 85
17.5.1 Configuration Files 85
17.5.2 Configuring SNMP Agent Port 85
17.5.3 Configuring Agent to Respond to SNMP v3 Requests 85
17.5.4 Configuring Agent Back to SNMP v1 86
17.5.5 Setting up SNMP v1 MIB Browser 86
17.5.6 Setting up an SNMP v3 MIB Browser 86
17.5.7 Changing the SNMP MD5 and DES Passwords 86
17.6 SNMP Traps 87
17.6.1 SNMP Trap Format 87
17.6.2 Proprietary SNMP Trap Format 87
17.6.3 Configuring SNMP Trap Format 88
17.6.4 Configuring the SNMP Trap Port 88
17.6.5 Configuring RSM to Send SNMP v3 Traps 88
17.6.6 Configuring RSM to Send SNMP v1 Traps 88
17.7 Configuring and Enabling SNMP Trap Addresses 89
17.7.1 Configuring SNMP Trap Addresses 89
17.7.2 Enabling and Disabling SNMP Traps 89
17.7.3 Alerts Using SNMP v3 89
17.8 Configuring SNMP Trap Acknowledgement 90
17.9 Configuring SNMP Trap Retries 90
17.10 Sending SNMP Traps for Unrecognized Events 90
17.11 Trap Connect Sensor 91
17.12 SNMP Security 91
17.12.1 SNMP v1 Security 91
17.12.2 SNMP v3 Security Authentication and Privacy Protocol 91
17.13 Additional Notes 92
17.13.1 Redundant ListDataItems MIB Objects 92

18.0 Remote Management Control Protocol 93
18.1 RMCP Client and Server Communication 93
18.2 RMCP Modes 93
18.3 Enabling and Disabling RMCP 94
18.4 RMCP Discovery 94
18.5 IPMB Slave Addresses 94
18.6 Communicating with RMCP Server on RSM 95
18.7 RMCP Security 95
18.7.1 RMCP User Privilege Levels 95
18.7.2 RMCP Maximum Privilege Levels 95
18.7.3 Configuring IPMI Command Privileges 95
18.7.4 BMC Key 96
18.7.5 Authentication 96
18.7.6 IPMI System GUID 96
18.8 RMCP over SCTP Transport 96

18.9 Supported IPMI Commands 97
18.10 Completion Codes for RMCP Messages 100

19.0 IPMI Pass-Through 101
19.1 Overview 101
19.2 Command Syntax 101
19.2.1 Command Request String Format 101
19.3 Response String 102
19.4 Usage Examples 102
19.4.1 Using the CLI 102
19.4.2 Using ShM API 102
19.4.3 Using SNMP 102

20.0 RSM Scripting 103
20.1 Command Line Interface Scripting 103
20.2 Event Scripting 103
20.2.1 Triggering Scripts from Health Events 103
20.2.2 Triggering Scripts from Event Codes 104
20.2.3 Script Execution 105
20.2.4 Listing Scripts Associated with Events 105
20.2.5 Disassociating Scripts from an Event 105
20.2.6 Script Synchronization 106
20.3 Environment Variables 106
20.4 Error Processing and Messages 107
20.4.1 Invalid pathname 107
20.4.2 Script does not exist 107
20.4.3 Pathname specified is a directory 107
20.4.4 Moved or removed script still associated with event 108
20.4.5 Script has zero bytes 108
20.4.6 Script lacks execute permission 108
20.4.7 Script is on the standby RSM 108
20.4.8 Unable to write to policy.conf 108
20.5 Default Scripts 108
20.6 Limitations 109
20.6.1 Usage of switchover commands 109

21.0 Operational State Management 110
21.1 Hot Swap States 110
21.2 Hot Swap Sensor 110
21.3 FRU Control Scripts 111
21.4 FRU Activation Policy 111
21.5 Checking Node Presence 111

22.0 Power Management 112
22.1 Node Operational Power Management 112
22.1.1 Power Levels 112
22.1.2 Shelf Power Budget 112
22.1.3 Power-on Sequence 112
22.2 Power Feed Targets 113
22.3 Forced Power State Changes on Blades 113
22.3.1 Powering Off a Blade 113
22.3.2 Powering On a Blade 113
22.3.3 Resetting a Blade 114
22.4 Obtaining the Power State of a Blade 114

23.0 Cooling and Fan Control 115
23.1 Temperature Condition Sensor 115
23.2 Cooling Policy 115
23.2.1 Process for modifying the shm.conf file 117
23.2.2 Normal Cooling Adjustments 117

23.3 Fan Control in Re-enumeration 118
23.4 Fan Tray Cooling Properties 118
23.5 Retrieving Current Cooling Level 118
23.6 Setting Current Cooling Level 118
23.7 Fan Tray Sensors 119
23.8 Control Modes for Fan Trays 119
23.8.1 RSM Control Mode 119
23.8.2 Fantray Control Mode 119
23.8.3 Emergency Shutdown Control Mode 119
23.9 Automatic Control Mode Change 120
23.10 Fan Tray LED 120

24.0 Electronic Keying Management 121
24.1 Point-to-Point EKeying 121
24.2 Bused EKeying 121
24.3 EKeying CLI Commands 121

25.0 CDMs, Shelf FRU, and FRU Information 122
25.1 Chassis Data Modules 122
25.2 Shelf FRU Election Process 122
25.3 Shelf FRU Information 122
25.4 FRU Information 122
25.4.1 Physical IPMC FRU 0 123
25.4.2 Virtual IPMC FRU 0 127
25.4.3 Virtual IPMC FRU 1 129
25.4.4 Virtual IPMC FRU 2 129
25.4.5 Virtual IPMC FRU 3 129
25.4.6 Virtual IPMC FRU 4 129
25.4.7 Virtual IPMC FRU 5 129
25.4.8 Virtual IPMC FRU 6 130
25.4.9 Virtual IPMC FRU 7 130
25.4.10 Virtual IPMC FRU 8 130
25.5 FRU Query Syntax 130
25.6 Shelf Address 132

26.0 Command and Error Logging 133
26.1 Log Levels and Facilities 133
26.1.1 Environment Variables 133
26.1.2 Log Level Control 133
26.2 Command Logging 134
26.3 Error Logging 134
26.3.1 error.log 134
26.3.2 debug.log 134
26.4 Linux* logger 135
26.5 Configuring syslog 135
26.5.1 Log Rotation and Archives 136
26.5.2 Restarting syslog-ng 136
26.5.3 Caveats and Limitations 136

27.0 Diagnostics 138
27.1 U-Boot Diagnostic Tests 138
27.1.1 BOARD_INIT_RAM_TEST 138
27.1.2 POST Diagnostics 138
27.1.3 Manufacturing Diagnostics 139
27.2 Run-Time Diagnostics 141
27.2.1 Flash Diagnostics 141
27.2.2 Ethernet Diagnostics 141
27.3 Reboot Reason Discovery 141
27.4 RSM Crash Logging 142

27.5 Core Dump 142
27.6 Kernel Crash Logging 143
27.6.1 Kinds of Data Logged 143
27.6.2 Accessing Logged Data 143
27.6.3 Kernel Crash Log Rotation 143
27.6.4 Sample Log File 143
27.7 cmmdump Utility 145
27.8 Operating System Flash Corruption Detection & Recovery 145
27.8.1 Monitoring Static Images 145
27.8.2 Monitoring Dynamic Images 145

28.0 Statistics 146
28.1 Querying Statistics Values 146
28.2 OS Statistics 147

29.0 Time Synchronization 148
29.1 Default Configuration 148
29.2 Configuring NTP Client 148
29.3 Configuring NTP Server 150
29.4 Configuring NTP Server in Broadcast Mode 150
29.5 Time Synchronization Sensor 151
29.6 RTC Synchronization 151
29.7 Configuration File 151

30.0 Setting Up the RSM 152
30.1 Connecting to the RSM 152
30.2 Initial Setup 152
30.2.1 Setting IP Address Properties 152
30.2.2 Setting a Hostname 153
30.2.3 Mounting NFS 153
30.2.4 Setting Time for Auto-logout 153
30.2.5 Setting Date and Time 153
30.2.6 Establishing an Interactive Session 154
30.2.7 Connect through SSH 154
30.2.8 Rebooting the RSM 155

31.0 IP Network Configuration 156
31.1 Introduction 156
31.2 Shelf Manager IP Connection Record 156
31.3 OEM Network Data Record 156
31.4 Startup Behavior 158
31.5 Setting and accessing network configuration data 158
31.5.1 Setting the Active Network Direction 159
31.5.2 Getting the Active Network Direction 159
31.5.3 Setting Data for Active RSM 159
31.5.4 Retrieving Data for Active RSM 160
31.5.5 Setting Ethernet Port Data 160
31.5.6 Retrieving Ethernet Port Data 161
31.5.7 Resetting Ethernet Port Data to Factory Default Values 161
31.6 Examples 162
31.6.1 Setting Active RSM Data 162
31.6.2 Setting eth0 Network Configuration Data for RSM1 162
31.6.3 Setting eth1 Network Configuration Data for RSM1 162
31.6.4 Setting eth2 Network Configuration Data for RSM1 163
31.6.5 Setting eth3 Network Configuration Data for RSM1 163
31.6.6 Querying Factory Defaults 164
31.7 Using ShM API to Set and Get Network Configuration Data 164
31.8 Using SNMP to Set and Get Network Configuration Data 164
31.9 Start-up Network Configuration Data 164

31.10 Synchronization Between RSMs 164
31.11 Setting Ethernet Bonding 164
31.11.1 Enabling/Disabling Ethernet Bonding 165
31.11.2 Bonding Configuration 165
31.11.3 Verifying Proper Bonding Operation 166
31.11.4 Bonding Tests 167

32.0 Updating RSM Software 168
32.1 Overview 168
32.2 Main Features of Firmware Update Process 168
32.3 Update Process Elements 168
32.4 Dual Image 168
32.4.1 Next Boot Role 169
32.4.2 Setting the Next Boot Role 169
32.4.3 Automatic Rollback 169
32.4.4 System Booting Failures 170
32.4.5 Restarting Specified Image 170
32.5 Critical Software Update Files and Directories 170
32.6 Generating the update package 171
32.7 Update Package 171
32.7.1 Update Package File Validation 172
32.7.2 Firmware Image Properties 172
32.8 Single RSM System 172
32.9 Redundant RSM Systems 172
32.10 CLI Software Update Procedure 172
32.11 Update Process 173
32.12 Local Upgrade Sensor 174
32.13 Configuration Upgrade 174
32.14 U-Boot Update Process 174

33.0 Chassis Component Firmware Update 175

34.0 FRU Update Utility 176
34.1 Overview 176
34.2 FRU Update Architecture 176
34.2.1 Required Files 176
34.2.2 Update Verification 176
34.2.3 FRU Data Recovery 177
34.3 FRU Update Usage 177
34.3.1 ipmitool Parameters 178
34.3.2 Chassis slot and FRU IPMB addresses 180
34.3.3 Command Examples: 180
34.4 Customizing FRU-Specific Data 181

35.0 Third-Party Chassis Integration 183
35.1 Introduction 183
35.2 Integrating RSM Firmware into Chassis 183
35.3 Creating Chassis FRU Information 183
35.3.1 About frugen.pl 183
35.3.2 Command Options 184
35.4 Creating Configuration Files 184
35.5 cmm.ini 185
35.5.1 IPMB Section 185
35.5.2 Alias Input Section 185
35.5.3 Alias Output Section 186
35.5.4 CMM Section 186
35.5.5 Blade Section 186
35.5.6 FanTray Section 187
35.5.7 PEM Section 187

35.5.8 Power Feed Section 187
35.5.9 Fan section 188
35.5.10 PEM Section 188
35.6 Installing Configuration Files 189
35.7 Adding Files to RSM 189
35.7.1 Copying Files to RSM Manually 189
35.7.2 Creating OEM.zip File 189
35.7.3 Adding Chassis Support using Update Command 190
35.8 Assumptions and Limitations 190
35.8.1 LED Control 190
35.8.2 Chassis Data Module 190
35.8.3 Sensors 191
35.8.4 Fronted FRU Aliasing 191

36.0 Agency Information 192
36.1 North America (FCC Class A) 192
36.2 Canada – Industry Canada (ICES-003 Class A) 192
36.3 Safety Instructions 192
36.3.1 English 192
36.3.2 French 193
36.4 Taiwan Class A Warning Statement 193
36.5 Japan VCCI Class A 193
36.6 Korean Class A 193
36.7 Australia, New Zealand 193

37.0 Safety Warnings 194
37.1 Mesures de Sécurité 195
37.2 Sicherheitshinweise 197
37.3 Norme di Sicurezza 198
37.4 Instrucciones de Seguridad 200
37.5 Chinese Safety Warning 202

A Sensor Numbers 203
A.1 Shelf Sensors 203
A.2 RSM Sensors 204
A.2.1 RSM Sensors - Physical IPMC 205
A.2.2 RSM Sensors - Virtual IPMC 208
A.2.3 Device Sensor Data Record (SDR) Repository 214

B IPMI Generic Sensor Events 215
B.1 Introduction 215
B.2 Explanation of Abbreviations and Symbols 215
B.3 Event Severity and Contribution to System Health 215

C IPMI Typed Sensor Events 221
C.1 Introduction 221
C.2 Explanation of Abbreviations and Symbols 221
C.3 IPMI Typed Sensor Tables 222

D OEM Sensor Events 244
D.1 Introduction 244
D.2 Explanation of Abbreviations and Symbols 244
D.3 PICMG Hot Swap Sensor 245
D.4 PICMG IPMB-0 Link Sensor 247
D.5 HA Trap Connect Sensor 248
D.6 HA Out of Service Request Sensor 249
D.7 HA In Service Request Sensor 249
D.8 HA State Sensor 250
D.9 DataSync Status Sensor 254
D.10 HA Health Score Sensor 255

D.11 HA Redundancy Sensor 256
D.12 HA Control Sensor 257
D.13 PMS Fault Sensor 259
D.14 PMS Info Sensor 260
D.15 PMS Health Sensor 261
D.16 Local Upgrade Sensor 262
D.17 Log Usage Sensor 264
D.18 Power Allocation Sensor 264
D.19 Power Budget Sensor 265
D.20 Cooling Policy Sensor 265
D.21 Temperature Condition Sensor 265
D.22 Re-enumeration Sensor 266
D.23 RT Diagnostics Sensor 267
D.24 Reboot Reason Sensor 268
D.25 Security Sensor 268
D.26 NTP Status Sensor 269
D.27 Non Compliant FRU Sensor 269
D.28 Filter Run Time Sensor 270
D.29 CMM Status Sensor 270
D.30 HA Peer Lost Sensor 272
D.31 Power Restoration Failure 273
D.32 IPMC Reset Sensor 273
D.33 LMP Reset Sensor 273
D.34 CFD Watchdog Sensor 273
D.35 IPMC HA State Sensor 274
D.36 IPMC Failover Sensor 274
D.37 System Firmware Progress Sensor 275

E Statistics 286
E.1 OS Statistics 286
E.2 Events Statistics 286
E.3 Data Synchronization Statistics 287
E.4 IPMI Generic Statistics 288
E.5 IPMI Message Pool Statistics 289
E.6 Cooling Statistics 289
E.7 Local Sensor Repository Statistics 290

F Legacy RPC Interface 291
F.1 Setting Up the RPC Interface 291
F.2 Using the RPC Interface 291
F.2.1 GetAuthCapability() 292
F.2.2 ChassisManagementApi() 293
F.2.3 ChassisManagementApi() threshold response format 300
F.2.4 ChassisManagementApi() string response format 300
F.2.5 ChassisManagementApi() integer response format 303
F.2.6 FRU String Response Format 304
F.3 RPC Sample Code 304
F.4 RPC Usage Examples 305

G Reference Information 308
G.1 AdvancedTCA* Product Information 308
G.2 AdvancedTCA Specifications 308
G.3 IPMI 308

H ShMgr Version Feature Differences 309
H.1 LISM 309
H.1.1 ShMgr software 7.1.x is designed to be a Location Independent Shelf Manager (LISM) 309
H.1.2 For version 8.x, the "software IPMC process" and associated functionality are decoupled from the LISM 309
H.2 Porting to version 8.1.X includes porting ShMgr software to a different platform 309
H.2.1 Wind River 3.0 309
H.2.2 New LMP processor 309
H.2.3 New IPMC 309
H.2.4 U-Boot firmware bootstrapping 309
H.3 Shelf management functionality is divided into two distinct components 309
H.3.1 Low-level code running on the Renesas H8S/2472 microcontroller (ShMC) 309
H.3.2 High-level code running on a Local Management Processor (LMP) 309
H.4 Cannot upgrade from ShMgr versions 5.2.x, 6.1.x, and 7.1.x 310
H.5 FRU power management 310
H.6 Performance improvements 310
H.6.1 Event management 310
H.6.2 SDR management 310

Chapter 1

1.0 Document Organization

1.1 Document Organization

This document describes the operation and use of the A6K-RSM-J shelf manager (RSM).

The following topics are covered in this document.

Chapter 2.0, “Introduction,” introduces the key features of the RSM. This chapter includes a product definition and a list of product features.

Chapter 3.0, “System Level Specifications,” provides system specifications for the RSM.

Chapter 4.0, “Front Panel LEDs,” describes LEDs.

Chapter 5.0, “Sensors,” defines sensors and access methods.

Chapter 6.0, “Health Events,” defines health events.

Chapter 7.0, “Alarms,” defines alarms and annunciators.

Chapter 8.0, “System Event Log,”specifies the content and architecture of System Event Log.

Chapter 9.0, “Trap Generation and Platform Event Filtering,” defines proprietary and IPMI methods for filtering platform events in the RSM.

Chapter 10.0, “High Availability,” specifies architecture and user instrumentation of high availability.

Chapter 11.0, “Re-enumeration,” describes chassis re-enumeration.

Chapter 12.0, “Process Monitoring and Integrity,” describes Process Monitoring service (PM) that monitors the general health of processes running on the RSM and takes recovery actions upon detection of failed processes.

Chapter 13.0, “Security,” specifies role based access control and user management in RSM.

Chapter 14.0, “Hardware Platform Interface,” gives brief description of HPI.

Chapter 15.0, “Shelf Management & OAM API,” gives brief description of OAM & ShM API.

Chapter 16.0, “Command Line Interface,” gives brief description of CLI.

Chapter 17.0, “Simple Network Management Protocol,” specifies how SNMP can be used for chassis management.

Chapter 18.0, “Remote Management Control Protocol,” specifies how RMCP and IPMI LAN interface can be used for chassis management.

Chapter 19.0, “IPMI Pass-Through,” specifies how IPMI Pass Through interface can be used for chassis management.

Chapter 20.0, “RSM Scripting,” specifies usage model for calling the Command Line Interface (CLI) indirectly through scripts using bash shell scripting.

Chapters 21.0 through 25.0 specify how RSM implements PICMG shelf management functions: operational state management, power and cooling management, E-Keys management, FRU and Shelf FRU information management.

Chapter 26.0, “Command and Error Logging,” describes RSM logging service.

Chapter 27.0, “Diagnostics,” specifies diagnostic instrumentation.

Chapter 28.0, “Statistics” specifies instrumentation for statistics.

Chapter 29.0, “Time Synchronization,” describes how RSM implements time management and synchronization.

Chapter 30.0, “Setting Up the RSM,” describes device setup and initial configuration.

Chapter 31.0, “IP Network Configuration,” describes how IP configuration is maintained and managed.

Chapter 32.0, “Updating RSM Software,” describes architecture and procedures of RSM firmware

Chapter 33.0, “Chassis Component Firmware Update,” addresses firmware update on other chassis components, such as fan trays, PEMs, etc.

Chapter 34.0, “FRU Update Utility,” describes the architecture and usage models of FRU Update utility.

Chapter 35.0, “Third-Party Chassis Integration,” describes how RSM must be configured in order to integrate into chassis from third party vendors.

Chapters 36.0 and 37.0 provide agency information and safety warnings.

Appendix A, “Sensor Numbers” lists the shelf and RSM sensor numbers, names and types.

Appendix B, “IPMI Generic Sensor Events” documents the generic sensors and their events that are implemented in the RSM firmware.

Appendix C, “IPMI Typed Sensor Events” documents the typed sensors and their events that are implemented in the RSM firmware.

Appendix D, “OEM Sensor Events” lists all of the OEM sensors and events defined for the RSM.

Appendix E, “Statistics” describes the statistics that are implemented in the RSM firmware.

Appendix F, “Legacy RPC Interface” describes how custom remote applications can administer the RSM by using remote procedure calls.

Appendix G, “Reference Information” provides links to data sheets, standards, and specifications for the technology designed into the RSM.

Appendix H, “ShMgr Version Feature Differences” describes the feature differences between the 8.x version of the A6K-RSM-J ShMgr software and earlier versions used on previous CMMs.

1.2 What’s New in This Manual

- Added a note to the +3.0V Battery sensor that event generation for the sensor is disabled when the RSM is used in an NECCH0001 chassis.
- The System Firmware Progress sensor table was moved from appendix C to appendix D because the sensor events are handled as OEM types, not IPMI types.
- Added section 34.2.3.1, shelf FRU data backup commands.
- Changes to documented output to match actual firmware output.
- RmcpProtocol command replaced with RmcpTransport.
- Event Logging Disabled sensor Assertion/Deassertion severity changed to OK for event codes 0x543, 0x544, and 0x545.
- Added sensors *CDM 1 Health* and *CDM 2 Health* to Table 76, Virtual FRU 1 and Virtual FRU 2.

1.3 Glossary of Terms Used in This Document

Table 1, "Glossary" lists a glossary of terms used in this document.

Table 1. Glossary (Sheet 1 of 2)

Term Used	Description
AdvancedTCA	Advanced Telecom Computing Architecture
AMC	AdvancedTCA* Mezzanine Card
ASCII	American Standard Code for Information Interchange
ATCA	Advanced Telecom Computing Architecture
CDM	Chassis Data Module
CLI	Command Line Interface
CRC	Cyclic Redundancy Check
DHCP	Dynamic Host Configuration Protocol
FFS	Flash File System
FIS	Flash Image System
FPGA	Field-Programmable Gate Arrays
FRU	Field Replaceable Unit
FTP	File Transfer Protocol
GPIO	General Purpose Input/Output
HPI	Hardware Platform Interface
HS	Hot Swap
IP	Internet Protocol
IPMB	Intelligent Platform Management Bus
IPMC	Intelligent Platform Management Controller
IPMI	Intelligent Platform Management Interface
LAN	Local Area Network
LED	Light Emitting Diode
LSB	Least Significant Bit
MIB	Management Information Base
MIB II	Management Information Base for Network Management II
MRA	MultiRecord Area
MSB	Most Significant Bit
OEM	Original Equipment Manufacturer
OS	Operating System
PEF	Platform Event Filtering
PEM	Power Entry Module
PICMG	PCI Industrial Computer Manufacturers' Group
RMCP	Remote Management Control Protocol
RPC	Remote Procedural Calls
RSM	Radisys Shelf Manager module
RTM	Rear Transition Module
SAF	Service Availability Forum
SBC	Single Board Computer
SDR	Sensor Data Record
SEL	System Event Log

Table 1. Glossary (Sheet 2 of 2)

Term Used	Description
SIF	Sensor Information File
ShMC	Shelf Management Controller
SNMP	Simple Network Management Protocol
SSH	Secure Socket Shell
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
WDT	Watchdog Timer

Chapter 2

2.0 Introduction

2.1 Overview

This document describes the features and specifications of the firmware and software that runs on the A6K-RSM-J Shelf Manager module (RSM). The A6K-RSM-J RSM is a shelf manager that monitors and controls the hardware components installed in an AdvancedTCA chassis.

The RSM plugs into a dedicated slot in compatible systems. It provides centralized management and alarming for up to 16 node and/or fabric slots as well as for system power supplies, fans, and power entry modules. The RSM may be paired with a backup RSM for redundant use in high-availability applications. In such a configuration one RSM functions as the active RSM and manages the devices in the chassis; the other RSM functions as a standby RSM, ready to take over management of the chassis if a failover is needed or requested.

The A6K-RSM-J has its own processor, memory, PCI bus, operating system, and peripherals. The RSM monitors and configures IPMI-based components in the chassis. When thresholds (such as temperature and voltage) are crossed or a failure occurs, the RSM captures these events, stores them in an event log, and sends SNMP traps. The RSM can query FRU information (such as serial number, model number, manufacture date, etc.), detect the insertion or removal of components (such as fan tray, CPU board, etc.), perform health monitoring of each component, control the power-up sequencing of each device, and control power to each slot via Intelligent Platform Management Interface (IPMI).

Note: This document assumes some basic familiarity with the Linux* operating system and associated tools (such as the `vi` text editor).

2.2 AdvancedMC* Support

The RSM firmware supports AdvancedMCs (Advanced Mezzanine Cards, or AMCs) as sub-FRUs on an SBC (Single Board Computer) or CPM (Compute Processing Module). This support includes power management of the AMCs, hot swap capability, and support for sensors on the AMC. The sensors can be read, the health of the AMC can be monitored and logged, and events pertaining to the AMC can be sent via SNMP traps. Scripts can be written to monitor the AMCs and take appropriate action in response to events generated by the AMC.

2.3 Third-party Chassis Integration

The A6K-RSM-J running version 8.1.x of the ShMgr firmware can be integrated into most shelves (chassis) that comply with the PICMG 3.0 Revision 2.0 (AdvancedTCA) specification. Provided with the proper configuration information, such as IPMB (Intelligent Platform Management Bus), topology, slot layout, hardware addresses, etc., the RSM firmware is able to manage most third party shelves that have been developed for the RSM hardware.

2.4 Specification Conformance

The RSM is designed to function in a chassis with components that conform to the PICMG* 3.0 Revision 2.0 AdvancedTCA* Base Specification, and the Intelligent Platform Management Interface Specification version 1.5 Document Revision 1.1, and version 2.0 Document Revision 1.0.

2.5 Related Documents

The following documents relate to the A6K-RSM-J shelf manager:

- *A6K-RSM-J Hardware Reference*
  Document Revision 0001, May 2011,
  Radisys
- *A6K-RSM-J Installation Guide*
  Document Revision 0001, May 2011,
  Radisys
- *A6K-RSM-J Firmware and Software Update Instructions*
  Document Revision 0004, June 2011,
  Radisys
- *Command Line Interface Reference for CMMs A6K-RSM-J, MPCMM0001, MPCMM0002*
  Document Revision 0002, January 2012
  Radisys
- *A6K-RSM-J, MPCMM0001 and MPCMM0002 Chassis Management Module ShM & OAM API Reference Manual*
  Document Revision 0001, August 2010,
  Radisys
- *Alert Standard Format Specification*
  Version 2.0, April 23, 2003
  Distributed Management Task Force, Inc.
- *Intelligent Platform Management Interface Specification v1.5*
  Document Revision 1.1, February 20, 2002
  Intel Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation
- *Intelligent Platform Management Interface Specification v2.0*
  Document Revision 1.0, February 12, 2004
  Intel Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation
- *Platform Management FRU Information Storage Definition*
  v1.0 Document Revision 1.1, September 27, 1999
  Intel Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation.
- *Platform Event Trap Format Specification*
  v1.0 Document Revision 1.0, December 7, 1998
  Intel Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation.
- *PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification*
  February 11, 2005
  PCI Industrial Computer Manufacturers Group
- *Service Availability Forum Hardware Platform Interface Specification*
  Version SAI-HPI-B.01.01, 2004
  Service Availability Forum
- *Service Availability Forum HPI-to-AdvancedTCA Mapping Specification*
  Version 0.9, July 2005
  Service Availability Forum
- *Alert Standard Format (ASF) Specification* version 2.0
  DMTF document DSP0136

- *RFC1057*
  Remote Procedure Call Protocol Specification
- *RFC1157*
  SNMPv1 message processing models
- *RFC1213*
  MIB II
- *RFC1215*
  SNMP TRAP v1
- *RFC1305*
  Network Time Protocol
- *RFC3410*
  SNMPv3
- *RFC3414*
  User-based Security Model
- *RFC3415*
  View-based Access Control Model (VACM)
- *RFC3416*
  SNMP TRAP v2
- *IPMI*
  Intelligent Platform Management Interface Specification
  Second Generation v2.0, Document Revision 1.0
  http://www.intel.com/design/servers/ipmi
- *PET*
  IPMI - Platform Event Trap Format Specification v 1.0
  http://www.intel.com/design/servers/ipmi
- *Appendix G, "Reference Information" on page 308*.

3.0 System Level Specifications

3.1 U-Boot*

The RSM enters into the U-Boot firmware to bootstrap the embedded environment once power is applied to the chassis.

3.2 Operating System

The RSM runs Wind River 3 on the FreeScale P2020 processor.

3.3 File System Organization

The general structure of the file system is like that of a typical UNIX* system. Table 2, “File System Organization” lists an outline of the file system organization. Not all directories are listed in this table, just those that are mount points or are otherwise important.

Table 2. File System Organization

Directory	Mounting point	Description
/	yes	Root of the file system
/bin	no	Major OS utilities
/sbin	no	Major OS administrative utilities
/dev	no	Kernel devices
/etc	yes	OS configuration
/etc/cmm	no	RSM configuration
/etc/cmm/chassis	no	Chassis specific configuration
/lib	no	OS libraries
/usr/bin	no	Additional OS utilities
/usr/lib	no	Additional libraries
/usr/cmm/bin	no	RSM binaries and other executables (e.g. tools)
/usr/cmm/lib	no	RSM dynamic libraries
/usr/local/data	yes	Crashdump storage area
/usr/share/cmm	no	User storage
/usr/share/cmm/bin	no	User executables
/usr/share/cmm/scripts	yes	User scripts
/var/log/cmm	yes	Log storage
/var/log/cmm/sel	no	System event log (incl. archives)
/var/log/cmm/cmm	no	RSM and OS error log files (incl. archives)
/var/log/cmm/cmm/crash	no	Crash log
/var/run	no	Symbolic link /tmp
/tmp	tmpfs	Temporary data in tmpfs
/proc	procfs	kernel info and control
/sys	sysfs	Kernel info

3.3.1 Flash Storage

RSM flash storage consists of two banks of 1 gigabyte each. The flash partitions and bank assignments are listed in Table 3.

Table 3. Flash Partitions and Bank Assignments

Partition	Bank Assignment
mtd0	Whole active flash bank
mtd1	Active flash bank U-Boot
mtd2	Active flash bank Linux
mtd3	Active flash bank raw persistent storage (should not be used)
mtd4	Whole backup flash bank
mtd5	Backup flash bank U-Boot
mtd6	Backup flash bank Linux
mtd7	Backup flash bank raw persistent storage (should not be used)
mtd8	Active flash bank JFFS persistent storage
mtd9	Backup flash bank JFFS persistent storage
mtd10	SPI boot flash active bank
mtd11	SPI boot flash backup bank

3.3.1.1 Whole Bank

This area contains the entire flash device, ignoring any partitioning.

3.3.1.2 U-Boot

This area contains space reserved for U-Boot applications.

3.3.1.3 Linux

This area contains the Linux kernel image and ramdisk image with RSM image and Linux root file system. The active RSM image is mounted at `/usr/cmm`.

3.3.1.4 Raw Persistent Storage

This area consists space used internally by the Linux kernel to provide persistent storage partitions.

3.3.1.5 JFFS File Systems

User executables and scripts are mounted at `/usr/share/cmm`. The scripts are located in the directory `/usr/share/cmm/scripts`.

Partition mounted at `/var/log/cmm` provides persistent storage for system event log (SEL), error logs, last reboot reason log, and other OS log files (incl. archives).

Variable system configuration is mounted at `/etc/cmm`. As the `/etc` directory is read-only (it is a part of the root file system), editable configuration files are located here and have symbolic links in `/etc`.

3.3.1.6 SPI Boot Flash

This area contains the U-Boot images and the U-Boot environment variables.

3.4 Random Access Memory

Total RAM size is 1 GB.

3.5 Configuration Files

The RSM configuration is stored in a number of configuration files in directory `/etc/cmm`. RSM configuration files use ASCII text format. The files and the parameters are described in the relevant sections of this Technical Product Specification.

When the RSM is running, user edits bypassing system management interfaces (e.g. CLI) are not allowed.

The following configuration files contain parameters corresponding to CLI dataitems: `shm.conf`, `policy.conf`, `trap.conf`, `snmpd.local.conf`, `rmcp.conf`, `ipmi.conf`, `timesync.conf`, `permissions.conf`, and `networks.conf`. When the RSM is running, the user can change a parameter value in one of these files by executing the proper CLI command.

Configuration files `snmpd.conf`, `pm.conf`, `events.conf`, and `busekey.conf` cannot be modified with CLI. The files *can* be edited by the user at any time. The new values are read once at RSM startup. File `local.conf` is writable by RSM but it should not be modified by the user.

Chassis configuration files are located in `/etc/cmm/chassis`. They are described in detail in Chapter 35.0, “Third-Party Chassis Integration” on page 183.

Note: If a given parameter is not present in a particular configuration file, it assumes the default value.

3.6 Factory Reset

The RSM startup script supports the factory reset command. When the user calls `cmm --factory-RESET`, all files located in directories `/etc/cmm`, `/var/log/cmm`, and `/usr/share/cmm/` are erased. Next, the erased configuration files and default scripts are replaced with factory default files stored in the read-only `/.etc-orig/cmm.skel` directory.

3.7 Application Hosting

The RSM allows applications to be hosted and run locally. This is useful for adding small custom management utilities to the RSM.

3.7.1 Startup and Shutdown Scripts

The RSM can run user-created scripts automatically on boot-up or shutdown. This can be done by editing the `/usr/share/cmm/scripts/startup` and `/usr/share/cmm/scripts/shutdown` files with a text editor. These files are standard shell scripts, so scripts can be added along with anything else that can be done in a shell script.

When `/etc/inittab` executes, it performs a typical sysvinit setup by calling each script in `/etc/rc.d/rc2.d` with a `start` argument. The script names match the format *`SDDscriptname`*, where *`DD`* is a two-digit number in increasing numerical order. Scripts are also provided for executing the `/usr/share/cmm/scripts/startup` files.

Note: At the time when a user-defined startup script is executed, the CLI may still not be available.

When the reboot command is executed from the shell prompt, that command in turn executes all scripts matching the format `/etc/rc.d/rc2.d/`*`KDDscriptname`*, where *`DD`* represents a two-digit number. These scripts are executed in increasing numerical order with a `stop` argument. The RSM software provides a script which calls the `/usr/share/cmm/scripts/shutdown` script, if it exists.

3.7.2 Available System Resources

Since the RSM has firmware of its own running at all times, user applications must adhere to certain resource and directory constraints to avoid disrupting the operation of the RSM firmware. Specifically, restrictions are placed on an application's consumption of file system storage space, RAM, and interrupts. Exceeding these guidelines may interfere with proper RSM operation.

3.7.2.1 Flash Storage

Applications should not perform excessive amounts of flash file I/O at runtime because this will impair performance of the RSM. The following directories are of interest:

`/usr/share/cmm/scripts` - Used for storing user scripts.

`/usr/share/cmm/bin` - Used for storing application binaries. This directory is not persistent.

The last two directories can comprise at most 1 MB of data.

3.7.2.2 RAM Disk Storage

Files in this location are stored in RAM and will be lost during RSM reboots. Due to the constraints of writing to flash memory, larger file operations such as decompressing an archive should be performed on RAM disk in the following directory: `/tmp`.

This directory is useful for storing temporary files. Applications should make a subdirectory for use with their temporary files. Do not add more than 5 MB of data to this location.

3.7.2.3 RAM Constraints

Up to 512 megabytes of RAM are available for user applications.

3.7.2.4 Interrupt Constraints

User applications should not use interrupts. All interrupts are reserved for use by the RSM firmware.

3.7.2.5 Priority Constraints

User applications must run with OS priority less than or equal to NORMAL.

3.8 System Management Interfaces

The following set of system management interfaces can be used by a remote System Manager application to manage the chassis:

- HPI
- Shelf Management & OAM API
- CLI
- SNMP
- IPMI over RMCP
- Legacy RPC

RSM supports Hardware Platform Interface (HPI) version B.01.01 [see *Service Availability Forum Hardware Platform Interface Specification*]. HPI is an industry standard interface defined by Service Availability Forum (SAF) to monitor and control highly available systems. The HPI allows user applications and middleware to access and manage hardware components via a standardized interface. HPI is covered in Section 14.0, “Hardware Platform Interface” on page 78.

RSM supports Shelf Management and OAM interface. The Shelf Management interface exposes functions defined as IPMI commands in accordance with*Intelligent Platform Management Interface Specification v2.0* and *PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification*. The remote OAM

interface defines new functions that cover functionalities not addressed in the above mentioned specifications, such as alarm management, upgrade, diagnostics, or performance measurements. Shelf Management & OAM API is covered in Section 15.0, “Shelf Management & OAM API” on page 79.

The Command Line Interface (CLI) connects to and communicates with the intelligent management devices of the chassis, boards, and the RSM itself. The CLI is an application that runs on top of the ShM and OAM API and can be accessed directly or through a higher-level management application. Administrators can access the CLI through Telnet or SSH. Using the CLI, users can access information about the current state of the system including current sensor values, threshold settings, recent events, and overall chassis health, access and modify shelf and RSM configurations, set fan speeds, perform actions on a FRU, etc. The CLI interface is covered in Section 16.0, “Command Line Interface” on page 81.

The chassis management module supports both queries and traps on Simple Network Management Protocol (SNMP) v1 or v3. A Management Information Base (MIB) for the entire platform is included with the RSM. The SNMP agent provides the support for the following MIBs:

- MIB II (RFC1213) - standard IETF MIB
- RSM MIB
- OAM MIB

The last two MIBs are RSM-related MIBs. SNMP agent sends unsolicited events received from RSM to the System Manager as SNMP traps. The traps are generated in IPMI Platform Event Trap format and RSM format. The traps are transmitted to the set of configurable recipients. SNMP is covered in Section 17.0, “Simple Network Management Protocol” on page 82.

Remote Management Control Protocol (RMCP) is a protocol that defines a method to send IPMI packets over a Local Area Network (LAN). The RMCP server on the RSM can decode RMCP packages and forward the IPMI messages to the appropriate destinations, including: SBC blades, power entry modules (PEMs), fan trays, and local destinations within the RSM. When there is a responding IPMI message coming from SBC blades, PEMs, or fan trays destined for the RMCP client, the RMCP server formats this IPMI message into an RMCP message and sends it to through the designated LAN interface back to originator. RMCP is covered in Section 18.0, “Remote Management Control Protocol” on page 93.

In addition to the HPI and ShM/OAM programmatic interfaces, the RSM can be administered by custom remote applications via remote procedure calls (RPC) legacy interface. With introduction of HPI and ShM/OAM API interfaces, the legacy RPC interface is deprecated and shall not be supported in the next firmware versions. The legacy RPC interface is covered in Appendix F, “Legacy RPC Interface” on page 291.

3.9 Ethernet Interfaces

The RSM has four Ethernet ports, with two ports positioned on the front faceplate and two provided through the connector on the backplane. All four Ethernet ports remain active. For configuration details, see Section 31.0, “IP Network Configuration” on page 156.

3.10 IPMB

An AdvancedTCA* Shelf uses an Intelligent Platform Management Bus (IPMB) for the management communication among all intelligent FRUs.

The sensors (Slot Ready) are maintained by the IPMC software.

3.11 Telco Alarms

Telco alarms provided on a system chassis can be used to announce system alarms. The RSM IPMC generates the Telco sensor events for major reset, minor reset, and cutoff for chassis types that have these input signals.

The power alarm, minor alarm, major alarm, and critical alarm can be controlled using the `Set Telco Alarm State` command. The IPMC illuminates the respective minor, major, and critical LEDs when the `Set Telco Alarm State` command is used to enable alarms.

4.0 Front Panel LEDs

The RSM has four LEDs on the front panel for displaying the status of the RSM. They include:

- One Power Good (PG) LED (Green)
- One Active (ACT) LED (Amber)
- One Out of Service (OOS) LED (Red or Amber)
- One Hot Swap (HS) LED (Blue)

For more information on the RSM LEDs, see the *A6K-RSM-J Shelf Manager Reference*.

4.1 LED Types and States

The RSM can retrieve values for LEDs on the RSM, fan trays, PEMs, and blades in the chassis. The following tables list the default values for the LEDs on the RSM. Other devices will likely have different LED properties that can be retrieved through the RSM. For information about LEDs on other devices, see the appropriate documentation for that device.

4.1.1 Power Good LED

The RSM maintains a power good LED to provide the health status of the RSM.

Table 4. RSM Power Good LED States

Color	Description
Off	No power to the RSM
Solid Green	Normal operation—power OK

4.1.2 Hot Swap LED

The RSM maintains a single blue hot swap LED to provide the status of the RSM itself. The Hot Swap LED cannot have its state set or changed; it is read-only.

Table 5. RSM Hot Swap LED States

Color	Description
Off	RSM is operational
Blinking	RSM is transitioning to or from an operational state
Solid Blue	RSM is not activated and can be safely extracted[1]

1. During the shutdown process, after the HS LED becomes solid blue, wait a few seconds before extracting the RSM board from chassis.

4.1.3 Active LED

The RSM maintains an active LED to indicate the operational status of the RSM.

Table 6. RSM Active LED States

Color	Description
Off	RSM is on standby
Solid Amber	RSM is active

4.1.4 Out of Service LED

The RSM maintains an out of service LED that shows the service status.

Table 7. RSM Out of Service LED States

Color	Description
Off	RSM is operating normally
Solid Red	RSM is out of service

4.2 Retrieving a Location's LED Properties

The properties of a location's LED control status can be retrieved using this command:

```
cmmget -l <location> -d ledproperties
```

4.3 Retrieving Color Properties of LEDs

The valid colors that an LED supports and the default color properties for that LED can be retrieved using the command:

```
cmmget -l <location> -t <led> -d ledcolorprops
```

Note: The above command does not accept the target `all_leds` or *n*`:all_leds` (where *n* is a sub-FRU ID) for the value of `<led>`.

4.4 Retrieving State of LEDs

The state of an LED on a location can be retrieved using the command:

```
cmmget -l <location> -t <led> -d ledstate
```

Note: The above command does not accept the target `all_leds` or *n*`:all_leds` (where *n* is a sub-FRU ID) for the value of `<led>`.

4.5 Using Lamptest Function

If you attempt the lamptest function with any device other than the shelf manager module itself, the RSM firmware will simply pass the request to that device. It is entirely up to the device to determine how to respond to or reject the request. If you attempt the lamptest function on the RSM, you must specify `all_leds`.

4.6 LED Boot Sequence

During the boot process, the LEDs change in a pattern as described in Table 8, "LED Event Sequence" to indicate boot progress. Once the RSM firmware is running, the administrator can control the LEDs through standard interfaces or via programmatic control.

Table 8, "LED Event Sequence" describes the sequence of events following the insertion of the RSM and the corresponding LED state for each event.

Table 8. LED Event Sequence

Event	Power Good LED	Hot Swap LED	Active LED	Out of Service LED
Initial insertion or power on with ejector latch closed	Off	Solid blue	Lit when the IPMC is the active shelf management controller (ShMC). Otherwise, the LED is off.	IPMC does not light this LED, but external software may control the LED using standard IPMI commands.
U-Boot* initialization	Solid green	Off		
U-Boot* initialization finished. User script running.	Solid green	Off		
Linux* initialization finished. OS at init level 1.	Solid green	Off		
RSM init script running. Core process loaded. RSM at M1	Solid green	Off		
Initial RSM initialization finished (FRU election). RSM at M2	Solid green	Off		
RSM IPMC at M3 or M4	Solid green	Off		

Chapter 5

5.0 Sensors

5.1 Overview

The shelf manager module recognizes and can log events from different sensor types as described in the *Intelligent Platform Management Interface Specification v1.5*. These sensors can be either threshold-based sensors or discrete sensors.

For more information on sensors and sensor types, see *Intelligent Platform Management Interface Specification v1.5*.

5.2 Threshold-based Sensors

Threshold-based sensors are those that generate or change an event status based on comparing a current value to a threshold value for a given hardware monitor device. Examples of threshold-based sensors are temperature, voltage, and fan tachometer sensors.

Threshold-based sensors generate events when a current value for a device becomes greater than or less than a given threshold value. The IPMI Specification defines six thresholds that can be assigned to a given sensor (see Figure 1, “IPMI Threshold Model” on page 31):

- Upper Non-Recoverable (UNR)
- Upper Critical (UC)
- Upper Non-Critical (UNC)
- Lower Non-Recoverable (LNR)
- Lower Critical (LC)
- Lower Non-Critical (LNC)

The sensor generates an event when its current reading rises above the upper thresholds or falls below the lower thresholds. The severity of the event generated depends on which threshold is crossed.

User can query sensor `<target>` for supported thresholds with a command:

```
cmmget -l <location> -t <target> -d thresholdsall
```

In order to learn selected threshold value, user must issue a command:

```
cmmget -l <location> -t <target> -d <threshold>
```

where `<threshold>` is one of supported threshold types.

5.2.1 Threshold-based Sensors on RSM

The shelf manager module maintains various voltage and temperature threshold sensors.

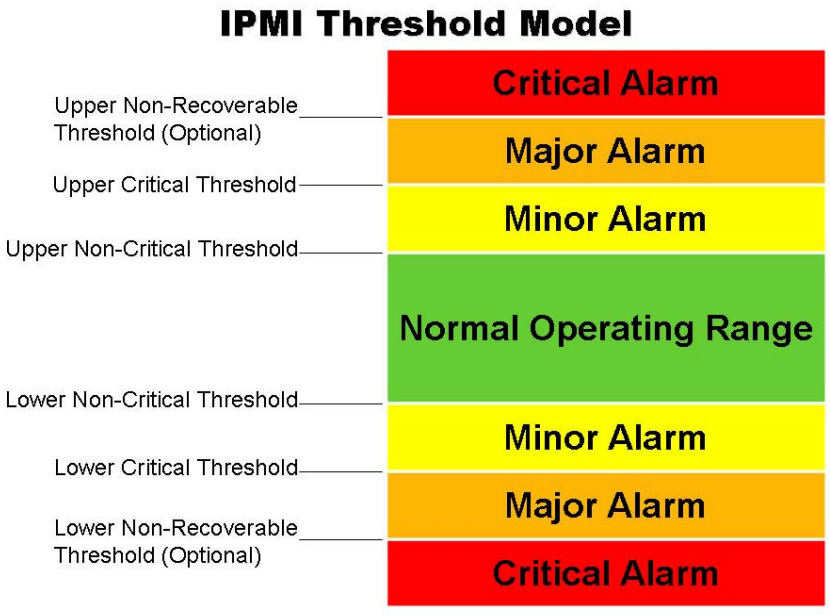
Table 9 shows the threshold type sensors present on the RSM, along with the Upper Non-Recoverable (UNR), Upper Critical (UC), Upper Non-Critical (UNC), Lower Non-Critical (LNC), Lower Critical (LC), and Lower Non-Recoverable (LNR) thresholds for each sensor.

Table 9. RSM Sensor Thresholds

Sensor Name (Sensor Number)	UNR	UC	UNC	LNC	LC	LNR
+12V (0Dh)	14.112	13.545	13.041	11.025	10.521	9.954
+3.6V I2C A (0Eh)	4.141	3.967	3.863	3.341	3.254	3.062
+3.6V I2C B (0Fh)	4.141	3.967	3.863	3.341	3.254	3.062
+3.3V (10h)	3.811	3.637	3.532	3.080	2.975	2.801
+3.0V Battery[a] (11h)	3.611	3.501	3.407	2.402	2.214	2.010
+2.5V (12h)	2.891	2.761	2.690	2.325	2.254	2.124
+1.8V (13h)	2.087	1.999	1.931	1.676	1.617	1.529
+1.2V (14h)	1.382	1.323	1.294	1.117	1.088	1.029
+1.05V CPU Core (15h)	1.215	1.168	1.121	0.991	0.944	0.897
+0.9V (16h)	1.050	0.991	0.979	0.838	0.814	0.767
CPU Temp (17h)	80	72	65	0	-5	-10
ADM1026 Temp (18h)	80	72	65	0	-5	-10
IPMC Temp (19h)	80	72	65	0	-5	-10

a. Event generation is disabled for the +3.0V Battery sensor when the RSM is used in an NECCH0001 chassis.

Figure 1. IPMI Threshold Model



5.3 Discrete Sensors

Discrete sensors are those that have a predefined finite set of states.

For example, the FRU Hot Swap sensor monitors the hot swap state of a FRU and is always in one of the predefined hot swap states: M1, M2, M3, M4, M5, M6, or M7.

Discrete sensors can generate events when the sensor makes a transition from one state to another. The severity of the event is determined by the RSM.

All discrete sensors can be queried for their current value. The value printed for discrete sensors is the bit vector of current assertions. The currently asserted states are printed in hexadecimal and followed by textual description.

For example:

```
bash# cmmget –l cmm –t "0:IPMI Version Change" –d current

The current value is 0x0008

in-service readiness state; active IPMI Version Change
```

5.3.1 OEM Sensors

OEM sensors are a special subgroup of discrete sensors where the discrete state information is specific to the OEM identified by the Manufacturer ID for the IPM device that is providing access to the sensor.

RSM maintains a number of OEM sensors. They are listed in Appendix D, “OEM Sensor Events”.

5.4 Sensor Event Description String

In response to an event generated by a sensor the RSM firmware outputs consistent event description strings for SEL entries, SNMP traps, and health events.

All sensor event description strings conform to the following syntax:

event_string: `Assertion` | `Deassertion, Event Code:` *event_code*

The event code has the format `0x`*NNNN*, where *N* is a hex digit. For example, the sensor description string for a processor IERR deassertion event looks like this:

```
Processor IERR detected: Deassertion, Event Code: 0x0220
```

An identical descriptive string is used for each pair of events: one for assertion and one for deassertion. The transition to asserted or deasserted is then indicated with the event direction “Assertion” or “Deassertion” following the descriptive string. The string terminates with the event code information.

For example:

```
Initial Data Synchronization complete: Assertion, Event Code: 0x1163
Initial Data Synchronization complete: Deassertion, Event Code: 0x1163
```

The first string asserts that initial data synchronization is complete. The second string deasserts this event. The event direction (`Assertion` or `Deassertion`) is applied to the same event description.

Note: The event code unambiguously identifies each distinct event.

The presence of the event code allows one to code scripts that key off of the numeric event code. This makes it unnecessary to parse the string beyond isolating the event code, which always appears in the same place in the string. Scripts written in this way will not be affected by any changes, corrections, or clarifications that might be made to the descriptive text portion of the string in future versions of the firmware, making such scripts easier to maintain.

Sensor event description strings and event codes are determined by RSM from event properties configuration maintained in `events.conf` configuration file. This topic is discussed in details in Section 6.4, “Health Event Property Configuration” on page 36.

For more information about scripting, see Section 20.0, “RSM Scripting” on page 103.

5.5 Sensor Information Details

Appendix B, “IPMI Generic Sensor Events,” lists all of the generic discrete sensors that the RSM recognizes. These sensors are taken from Table 36-2 of the IPMI Specification. The appendix includes event, string, event codes and the health contribution for each event associated with a given sensor.

Appendix C, “IPMI Typed Sensor Events,” lists all of the typed sensors that the RSM recognizes. These sensors are taken from Table 36-3 of IPMI Specification. The appendix includes event string, event codes and the health contribution for each event associated with a given sensor.

Appendix D, “OEM Sensor Events,” lists all of the Radisys OEM sensors that the RSM recognizes. The appendix includes event string, event codes and the health contribution for each event associated with a given sensor.

5.5.1 SEL Entries

Sensor events are recorded in the SEL. The SEL entry format is defined in Section 8.3, “SEL Display Format” on page 39.

5.5.2 SNMP Traps

SNMP traps are sent for events. The syntax of SNMP trap is defined in Section 17.6, “SNMP Traps” on page 87.

5.6 Sensor Targets

Available sensors for a location can be retrieved using the `listtargets` dataitem with the `cmmget` command.

For example, to view a list of sensor targets on the RSM, execute the following command:

```
cmmget -l cmm -d listtargets
```

The list of targets for the `cmm` location and the list of targets for the chassis location can be found in the *Alert Standard Format (ASF) Specification version 2.0*.

For complete lists of sensors on other components (for example, voltage sensors on a blade), see the Technical Product Specification (or equivalent document) for that product.

Chapter 6

6.0 Health Events

6.1 Overview

A health event (two words) refers to any generated system event that reports the state of a sensor and contributes to the overall health of the system.

See Section 5.0, “Sensors” on page 30 for more information on the different types of sensors (which are specified in the CLI as targets) that can generate events.

Note: The single word “healthevents” refers specifically to the `healthevents` dataitem or the output of that dataitem (results of a healthevents query). For more information on using the `healthevents` dataitem, see *Alert Standard Format (ASF) Specification version 2.0*.

Sensor names used in the command samples are for example only and may not be actual sensors.

6.2 Health Queries

The health of a particular location can be queried with this command:

```
cmmget -l <location> -d health
```

If `<location>` has no health problems, the output is:

location `has no problems`

On the other hand, if *location* has some problems, the output is:

location `has minor/major/critical events`

Setting *location* to `system`, the overall system health can be queried.

6.3 Healthevents Queries

Active health events for a particular target associated with a particular location can be viewed by executing a `healthevents` query to produce a health events listing as follows:

```
cmmget -l <location> -t <target> -d healthevents
```

Active health events are also displayed when `healthevents` queries are executed over SNMP. In addition, all health events are logged in the SEL and sent out as SNMP traps.

Note: SEL entries and SNMP traps do not include the severity of the event. Only the results of a `healthevents` query in the CLI display the severity of an event.

The following is the syntax of a string returned by a `healthevents` query for an associated active health event. The \n denotes a newline character.

timestamp\n
severity `Event : \t`*target health_event_string*`:` *event_direction*`, Event Code :` *event_code*\n

- *timestamp* is in the format *day month date hh*`:`*mm*`:`*ss year* (for example, Thu Dec 11 22:20:03 2006).
- *severity* is `Minor`, `Major`, or `Critical`.
- *target* is the name of the target with the sub-FRU ID prepended.
- *health_event_string* is a string describing the event. The content and the method of defining the event description string is described below in this chapter.
- *event_direction* is `Assertion` or `Deassertion`.
- *event_code* is `0x`*NNNN*, where each *N* is a hexadecimal digit.

For example:

```
bash# cmmget -l chassis:0 -t "0:CDM 2" -d healthevents

Thu Jan  5 15:15:37 2006

 Major Event : 0:CDM 2 Entity Absent: Assertion, Event Code : 0x0391
```

Note: Health events with a severity of OK may be displayed in a healthevents query for a limited time when they are asserted.

6.3.1 Healthevents Queries for Individual Sensors

Executing a healthevents query on a particular sensor target returns all active healthevents for that sensor target in a concatenated string. One sensor may have multiple events. For example, running the following healthevents query on a sensor:

```
cmmget -l cmm -t "<sensor name>" -d healthevents
```

might return multiple events that are active on the sensor in a concatenated string like this:

```
Mon Feb 2 19:51:05 2004

Major Event : CMM1:0:<sensor name> RTC Not working, Event Code : 0x007E

Mon Feb 2 19:51:09 2004

Major Event : CMM1:0:Both Etherent interfaces are not working, Event Code :
0x0080
```

6.3.2 Healthevents Queries for All Sensors on Location

You can execute a healthevents query on the `cmm` location in the CLI without specifying a target as follows:

```
cmmget -l cmm -d healthevents
```

This command returns all healthevents for all RSM sensors in a concatenated string. This includes all LAN, Voltage, and Temp sensors on the RSM. This ability to retrieve all healthevents on a location also applies to the `chassis`, `blade`*N*, `Fantray`*N* and `Pem`*N* locations.

6.3.3 No Active Events

When a healthevents query is executed in the CLI on a target that has no active events, a string is returned that is a single line with no timestamp or severity as follows:

target `has no problems.`

Only this string is returned; it is not concatenated with any other strings.

For example, assume that the following command is executed:

```
cmmget -l cmm -t "0:CPU Temp" -d healthevents
```

The following message is returned if the Brd Temp sensor has no active health events:

```
0:brd temp has no problems.
```

Executing a healthevents query through SNMP on a target with no active events returns different values than the CLI. When a healthevents query is executed using SNMP for a location or a target that has no active events (such as the cmmHealthEvents object), the value returned is a zero length string.

6.3.4 Not Present or Non-IPMI Locations

Executing a healthevents query of a blade or power supply (PEM) that is not present, or a target on a blade or power supply that is not present, returns an error if an empty slot is queried. If a blade is queried that is present but does not support IPMI, the message “Non IPMI Blade.” displays.

6.4 Health Event Property Configuration

Health event properties are configurable. They are maintained in the `/etc/cmm/events.conf` configuration file. Each event entry defines a number of properties, such as:

- System health contribution flag
- Health score weight multiplier

7.0 Alarms

7.1 Overview

An occurrence of a health event assigned to severity minor, major, or critical raises an alarm in the system. Active alarms are announced with annunciators.

7.2 Annunciators

Alarms are announced on annunciators and can be acknowledged by the user. A separate kind of alarm announcements are SNMP traps.

7.3 Acknowledging Alarms

An active alarm can be acknowledged (cleared) by the user. To clear all minor alarms in the system, enter this request:

```
cmmset -l system -d clearminor -v 1
```

This command affects the major alarm LED:

```
cmmset -l system -d clearmajor -v 1
```

A critical alarm cannot be cleared in that way; they are cleared when the reason for the alarm disappears.

Chapter 8

8.0 System Event Log

The RSM implements a System Event Log (SEL) in accordance with Section 3.5 of "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification".

When a system event is recorded in the RSM's system event log, it contains 16 bytes. The meaning of the bytes is specified in Table 26-1 in "Intelligent Platform Management Interface Specification v1.5". The RSM firmware uses the 16 bytes of data from a SEL entry to produce human readable output. If the firmware does not have enough encoded knowledge to translate the event, the firmware handles it as an unrecognized event. For instance, an event with Record Type of OEM timestamped or non-timestamped is treated as an unrecognized event. A standard IPMI event is also treated as an unrecognized event if it is not supported by the firmware translation code.

The RSM can display and trap both recognized and unrecognized events.

8.1 SEL Architecture on RSM

The RSM SEL is implemented as one master file `sel.dat` and a number of archives. All SEL files are stored locally in the `/var/log/cmm/sel` directory. The SEL contains a list of all sensor events in the chassis.

The SEL capacity is configurable. In order to keep the SEL from overflow, which causes loss of event logging, the SEL size is monitored by the RSM. The RSM implements the "Log Usage" Sensor and provides a default policy associated with this sensor event. If SEL size reaches 95% of configured capacity, the current SEL master file is closed, archived, and saved in the directory `/var/log/cmm/sel`. The names of the saved archives are `sel.dat.N`, where N is the number of the SEL archive.

The content of the SEL archive is limited by two parameters: the maximum total size of the archive and the maximum number of archived files. Once any of these limits is reached, the process rolls over and begins overwriting the oldest archives.

Caution: Archived files should *never* be decompressed on the RSM. The resulting prolonged writing to the flash file can disrupt the operations of the RSM. Instead, transfer the files using FTP to a different computer or system and decompress the archive there using an appropriate utility (such as `gzip`).

For a detailed description of "Log Usage" sensor, refer to Appendix D, "OEM Sensor Events".

8.2 Retrieving SEL

To retrieve a SEL from the RSM, execute the following CLI command:

```
cmmget -l <location> -d sel
```

The *location* parameter on a chassis can be any one of the following: `cmm`, `chassis`, `blade`*N* , `FanTray1`, `FilterTray1`, `PEM1`, or `PEM2`. The *location* parameter can also be followed by a FRU ID to retrieve only SEL entries for the specified sub-FRU.

The `cmmget` command filters the SEL entries and returns only events associated with the specified location. Certain individual FRUs (such as blades) may keep their own local SELs that can also be retrieved with the `cmmget` command.

Note: The available locations will depend on the configuration of the specific chassis.

8.3 SEL Display Format

When you list the contents of the SEL with the `cmmget` command, the format for each displayed SEL entry has three possible parts: the header, the translated text, and the raw output.

8.3.1 Header

The first part of SEL entry is a standard header. It consists of the timestamp followed by a newline \n character.

timestamp\n

timestamp is displayed in one of these two forms:

- A SEL event that has a timestamp (recognized System Event Records and OEM timestamped events) in the format [Day] [Month] [Date] [HH:MM:SS] [Year]. For example, Thu Apr 14 22:20:03 2005.
- OEM non-timestamped sensors, which display the text `Date/time unknown`.

8.3.2 Text Translation

The next portion of the SEL entry can be enabled or disabled as described later in this section. This provides the text interpretation of the event. Its format is shown below:

\t*location*\t*sensor_name*\t*health_event_string*`:` *event_direction*`, Event Code :` *event_code*\n

where

- *location* is the device where the sensor *sensor_name* is located
- *sensor_name* is the name given to the sensor in the Sensor Data Record (SDR).
- *health_event_string* is a string describing the event. The content and the method of defining the event description string is described in Chapter 5.0, “Sensor Event Description String” on page 32.
- *event_direction* is `Assertion` or `Deassertion`.
- *event_code* is `0x`*NNNN*, where each *N* is a hexadecimal digit.
  \t' stands for a Tab character, and '\n' for newline.

8.3.3 Raw Output

The final portion that a SEL entry might contain is the “raw” portion of the trap. This reports the original sixteen bytes of the system event as ASCII, upper case, hex bytes.

For example:

```
\tRaw Hex : [ 12 34 56 78 9A 0C 33 81 F2 1B 39 42 DE 64 BA 88 ]\n\n
```

At the end of the SEL display, there are always two trailing newlines (denoted by \n). '\t' stands for a Tab character.

Note: There is a space immediately after the open bracket and immediately before the close bracket. This is intended to make parsing the string easier.

8.3.4 Configuring SEL Display Format

The dataitem SelFormat controls whether the “text” portion or the “raw” portion of the SEL entry is displayed in addition to the header (which is always displayed). To configure the SEL format, execute the command:

```
cmmset -d selformat -v <format>
```

where *format* is one of the above:

- 1 - text
- 2 - raw
- 3 - text & raw

See 8.3.4.1 through 8.3.4.3 for details.

To retrieve the configured SEL display format execute cmmget on this dataitem.

Note: The sixteen bytes of raw hex data shown are an example of the display format. The actual data will be different.

Note: '\t' stands for a Tab character, and '\n' for newline.

8.3.4.1 selformat = 1 (text)

If SelFormat is set to 1 (text), the output is header plus text. The output will look as follows:

timestamp\n

\t*location*\t*sensor_name*\t*health_event_string*: *event_direction*, Event Code : *event_code*\n

8.3.4.2 selformat = 2 (raw)

If SelFormat is set to 2 (raw), the output is as shown below. The raw format is useful for scripting. Scripts can also use the command: cmmget –l <location> –d rawsel to obtain raw SEL information.

```
timestamp\n
\tRaw Hex : [ 12 34 56 78 9A … (16 bytes hex) ]\n\n
```

8.3.4.3 selformat = 3 (text & raw)

If SelFormat is set to 3 (text & raw), the output is as shown below:

timestamp\n

\t*location*\t*sensor_name*\t*health_event_string*: *event_direction*, Event Code : *event_code*\t*Raw Hex* : [12 34 56 78 9A … (16 bytes hex)]\n\n

8.3.5 Displaying Unrecognized SEL Events

If the dataitem SelDisplayUnrecognizedEvents is set to 1, the RSM displays unrecognized events. Otherwise, the RSM does not display unrecognized events. The default value stored in the configuration file is 0.

8.4 Retrieving SEL in Raw Format

To retrieve the SEL in its raw format execute the following CLI command:

```
cmmget -l <location> -d rawsel
```

8.5 Clearing SEL

The following CLI command clears the SEL on the RSM:

```
cmmset -l cmm -d clearsel -v clear
```

Caution: This command clears the SEL on both the active and standby RSM. Since the RSMs use a single flat file to store events, this command clears *all* events in the SEL and moves them into the archive.

8.6 SEL Configuration

SEL capacity specifies the maximum number of entries that one SEL master file can comprise. It can be configured with CLI command:

```
cmmset -l cmm -d selcapacity -v <capacity>
```

SEL capacity must be greater or equal to the value of the minimal SEL capacity parameter stored in the configuration file `/etc/cmm/shm.conf`.

Note: Changes of SEL capacity apply to the next SEL instance, not the currently opened one.

To get SEL capacity, execute the command:

```
cmmget -d selcapacity
```

The command returns the capacity for the currently opened SEL file, the configured capacity (they may differ), and the current SEL file occupancy.

To get the configuration of the SEL archive maintained in non-volatile storage, execute the CLI command:

```
cmmget -l cmm -d selArchiveInfo
```

The command returns the maximum number of SEL archive files and the maximum total size of SEL archives in kilobytes maintained in non-volatile storage.

The latter parameter is configurable with this CLI command:

```
cmmset -l cmm -d selarchivesize -v <size>
```

where <size> denotes the maximum total size of SEL archives in kilobytes. Value 0 means an unlimited size for the SEL archive. In this case, other limitations apply to the SEL archive, such as the maximum number of SEL archive files or the amount of free non-volatile storage space.

All SEL parameters are stored in the `/etc/cmm/shm.conf` configuration file.

Chapter 9

9.0 Trap Generation and Platform Event Filtering

9.1 Trap Generation and Platform Event Filtering

The RSM can generate SNMP Traps based on every Platform Event and every SEL entry This includes entries logged via the standard “Add SEL Entry” IPMI command, with any SEL Record Type, including OEM SEL Type.

The RSM generates SNMP Traps using Platform Event Filtering, based on the “Intelligent Platform Management Interface Specification v2.0” specification. For support details refer to Chapter 9.3.

Platform Event Filtering has the following configuration interface:

- CLI/RPC; for CLI command details, refer to Chapter 16.0, “Command Line Interface”
- SNMP
- Shelf Management & OAM API; for details, refer to Chapter 15.0, “Shelf Management & OAM API”

Platform Event Filtering can be configured using IPMI commands. For support details, refer to Chapter 9.3. For command details, refer to “Intelligent Platform Management Interface Specification v2.0”.

9.2 Configuration

The following section describes how to configure trap generation and Platform Event Filtering. The description is based on CLI commands. The PEF configuration parameters are based on the “Intelligent Platform Management Interface Specification v2.0” specification. For parameter description details, refer to “Intelligent Platform Management Interface Specification v2.0” unless otherwise specified.

The following elements can be configured for trap generation and Platform Event Filtering:

- Event Filtering Method; The method can be “legacy” or “pef”
- PEF Filter; The RSM maintains a table of filters. The table is indexed in the range <1-128>. Each filter defines certain matching rules. If an event matches the specified rule, an action is triggered. Only the “Send Alert” type of action is supported.
- PEF Alert Policy; The RSM maintains a table of alert policies. The table is indexed in the range <1-128>. An alert policy defines a destination to which a trap will be sent and alert string matching rules.
- PEF Alert String: The RSM maintains a table of alert strings. The table is indexed in the range <1-255>. The alert string is sent as a content of a trap.
- System GUID; This is the GUID value that is sent in a trap

9.2.1 Event Filtering Method

The following command gets the configured filtering method.

```
cmmget –d PefEventFilteringMethod
```

The following command sets the filtering method:

```
cmmset –d PefEventFilteringMethod –v <method>
```

9.2.2 PEF Filter

There can be up to 128 filters configured.

The following command template is used to configure a PET filter.

```
cmmset –t PefFilter:<index> -d <data item> –v <value>
```

The following data items can be configured for each filter:

- Status; this parameters defines if a filter is enabled or disabled
- Policy; Alert Policy Number for this filter
- Severity; Event Severity
- SlaveAddress; event Slave Address
- LUN; event LUN
- SensorType; Sensor Type
- SensorNumber; Sensor #
- EventType; Event/Reading Type
- EventOffsMask; Event Data 1 Event Offset Mask
- DataAndMask; this is a 48 bit mask consisting of:
  {Event Data 1 AND Mask, Event Data 2 AND Mask, Event Data 3 AND Mask}
- DataCmp1; this is a 48 bit mask consisting of:
  {Event Data 1 Compare 1, Event Data 2 Compare 1, Event Data 3 Compare 1}
- DataCmp2; this is a 48 bit mask consisting of:
  {Event Data 1 Compare 2, Event Data 2 Compare 2, Event Data 3 Compare 2}

For example, the following command configures a slave address for a PET filter number 120:

```
cmmset –t PefFilter:120 –d SlaveAddress –v 40
```

This example shows the usage of the command retrieving the current filter configuration:

```
cmmget –t PefFilter:120 –d Show

PefFilter:120

Status:                                 enabled

Policy Number:                          10

Severity:                               1

Slave Address:                          40

LUN:                                    1

Sensor Type:                            10

Sensor Number:                          100

Event Type:                             10

Event Offset Mask:                      0x00FF

AND Mask for Event Data:                0x00FFFF

Compare 1 Mask for Event Data:          0x00FF00

Compare 2 Mask for Event Data:          0x00F0F0
```

9.2.3 PEF Alert Policy

There can be up to 128 alert policies configured.

The following command template is used to configure an alert policy:

```
cmmset –t PefAlertPolicy:<index> -d <data item> –v <value>
```

The following data items can be configured for each alert policy:

- Status; this parameters defines if a policy is enabled or disabled
- Number; Alert Policy Number
- Rule
- Destination; one of five SNMP trap destinations
- StringLookup; string lookup method, which can have a value eventSpecific or notEventSpecific
- eventSpecific; the conjunction of String Selector and Event Filter Number is used to perform Alert String lookup
- notEventSpecific; the String Selector is used to perform Alert String lookup
- StringSelector; String Selector (Alert String Set)

For example, the following command configures a string lookup method for an alert policy number 20:

```
cmmset –t PefAlertPolicy:20 –d StringLookup –v eventSpecific
```

This example shows the usage of the command retrieving the current policy configuration:

```
cmmget –t PefAlertPolicy:120 –d Show

PefAlertPolicy:120

Status:                 enabled

Policy Number:          10

Policy Rule:            always

Destination Id:         2

String Lookup Method:   eventSpecific

String Selector:        1
```

9.2.4 PEF Alert String

There can be up to 255 alert strings configured.

The following command template is used to configure an alert string:

```
cmmset –t PefAlertString:<index> -d <data item> –v <value>
```

The following data items can be configured for an alert string:

- SetNumber; Alert Set Number
- FilterNumber; Filter Number
- String

For example, the following command configures a slave address for alert string number 14:

```
> cmmset –t PefAlertString:14 –d String –v “Sample alert string”
```

The following example shows the usage of the command retrieving the current alert string configuration:

```
cmmget –t PefAlertString:14 –d Show

PefAlertString:14

Set Number:                    1

Event Filter Number:           10

Alert String:                  “Sample Alert String”
```

9.2.5 System GUID

There are two possible system GUID sources:

- static; the GUID is configured using CLI
- command; this is the same GUID as returned by Get System GUID IPMI command.

The following command gets the configured system GUID source.

```
cmmget –d PefSystemGuidSource
```

The following command sets the system GUID source:

```
cmmset –d PefSystemGuidSource -v <source>
```

If the system GUID source is set to “static” the following command sets the required value.

```
cmmset –d PefSystemGuid –v <guid>
```

If the system GUID source is set to “command”, the GUID cannot be set with CLI command.

9.3 Supported PEF Functionality

The below tables specify which PEF features are implemented with respect to the "Intelligent Platform Management Interface Specification v2.0" specification.

Table 10. PEF functionality support

PEF feature	Comment
Power Down, Power Cycle, Reset, Diagnostics Interrupt actions	This feature is not supported.
Deferred Alert Processing	This feature is not supported. This feature is useful only when alerts are sent over communication channels on which one alert can block sending other alerts (for example modem callbacks). RSM does not support generating alerts other than SNMP trap messages sent over LAN.
PEF Postpone Timer	This feature is not supported. This feature is only useful when PEF is implemented on an IPMC associated with a payload processor. In such case, the postpone timer is used to let the payload processor the possibility to handle events before PEF is applied.
PEF Startup Delay	This feature is not supported. This feature applies only in conjunction with Power Down, Power Cycle and Reset actions.
Logging of PEF Actions to SEL	This feature is not supported.

The tables here specify which PEF IPMI commands and configuration parameters are defined in "Intelligent Platform Management Interface Specification v2.0" are supported.

Table 11. PEF IPMI commands support

PEF Command	Comments
Get PEF Capabilities	Always indicates that only 'Alert' action is supported
Arm PEF Postpone Timer	Not supported
Set PEF Configuration Parameters	See Table 1-3 for the list of supported parameters
Get PEF Configuration Parameters	See Table 1-3 for the list of supported parameters
Set Last Processed Event ID	Not supported
Get Last Processed Event ID	Not supported
Alert Immediate	Not supported

Table 12. Supported PEF configuration parameters

Parameter Selector	PEF Configuration Parameter	Comment
0	Set In Progress	Rollback not supported
1	PEF Control	Only bit 0 can be set. All other bits must always be zero (both in Get and Set operation). When PEF is disabled, SNMP Trap Generator uses Legacy Filtering.
2	PEF Action global control	Only 'enable Alert' action supported
5	Number of Event Filters	Fully supported
6	Event Filter Table	Fully supported
7	Event Filter Table Data1	Fully supported
8	Number of Alert Policy Entries	Fully supported
9	Alert Policy Table	Fully supported
10	System GUID	Fully supported
11	Number of Alert Strings	Fully supported
12	Alert String Keys	Alert String 0 not supported (no support for Alert Immediate command)
13	Alert Strings	Alert String 0 not supported (no support for Alert Immediate command)
96	SEL Filter Entry	[7] – Reserved [6:0] - PEF filter entry to be used to process OEM SEL Records. If the field is 00h, no PEF action is started for OEM SEL Records.

9.4 PET Trap

The RSM constructs trap messages in PET format both for SEL Event Records and OEM SEL Records. "Platform Event Trap Format Specification" defines the trap format only for SEL Event Records. The trap format for OEM SEL Records is similar to the format defined in "Platform Event Trap Format Specification" with the exceptions:

- Some fields that are not valid for OEM SEL Records are set to an arbitrary selected value,
- A raw SEL entry is appended to the OEM Custom Fields with Record Type equal to 3h and Record Encoding equal to 00b (binary).

Table 13, "PET Trap for SEL Event and OEM SEL Event" presents details about how a PET trap is constructed.

Table 13. PET Trap for SEL Event and OEM SEL Event

PET Field	Value for SEL Event Record	Value for OEM SEL Event
enterprise	.1.3.6.1.4.1.3183.1.1	
agent-addr	Network Address	
generic-trap	EnterpriseSpecific(6)	
Timestamp	host-uptime	
engineID (for SNMPv3)	0x0102030405	
Authentication protocol (for SNMPv3)	MD5	
Privacy protocol (for SNMPv3)	DES	
Specific Trap		
Sensor Type	From SEL Event Record	00h
Event Type	From SEL Event Record	00h
Event Offset	From SEL Event Record	00h
Variable Bindings		
GUID	According to pet_system_guid_source parameter	
Sequence Number	Internal counter	
Local Timestamp	From SEL Event Record	From OEM SEL Record if the record is timestamped. 00000000h – otherwise
UTC Offset	From Operating System	
Trap Source	20h	
Event Source Type	20h	
Event Severity	From PEF Event Filter Entry (for PEF filtering) or from Alarm Monitor API (for Legacy Filtering)	
Sensor Device	From SEL Event Record	FFh
Sensor Number	From SEL Event Record	FFh
Entity	From SDR Repository Manager	0h
Entity Instance	From SDR Repository Manager	0h
Event Data	From SEL Event Record	All zeros
Language Code	FFh (unspecified)	
Manufacturer ID	343 (Intel Corporation)	
System ID	Product ID retrieved using “Get Device ID” command sent to local IPMC	
OEM Custom Fields	Alert String (for PEF filtering) or Health Event String (for Legacy Filtering)	Alert String (for PEF filtering) or Health Event String (for Legacy Filtering) Additionally whole SEL record as Record Type equal to 3h and Record Encoding equal to 00b (binary).

10.0 High Availability

10.1 Overview

The RSM supports redundant operation with automatic failover in a chassis using redundant RSM slots. In systems where two RSMs are present, one acts as the active and the other as the standby[1]. Both RSMs monitor each other, and either one can trigger failover if necessary.

Data from the active RSM is synchronized to the standby RSM whenever any changes occur. Data on the standby RSM is overwritten. A full synchronization between active and standby RSMs occurs on initial power up, or any insertion of a new RSM.

The active RSM is responsible for shelf FRU information management when RSMs are in redundant mode.

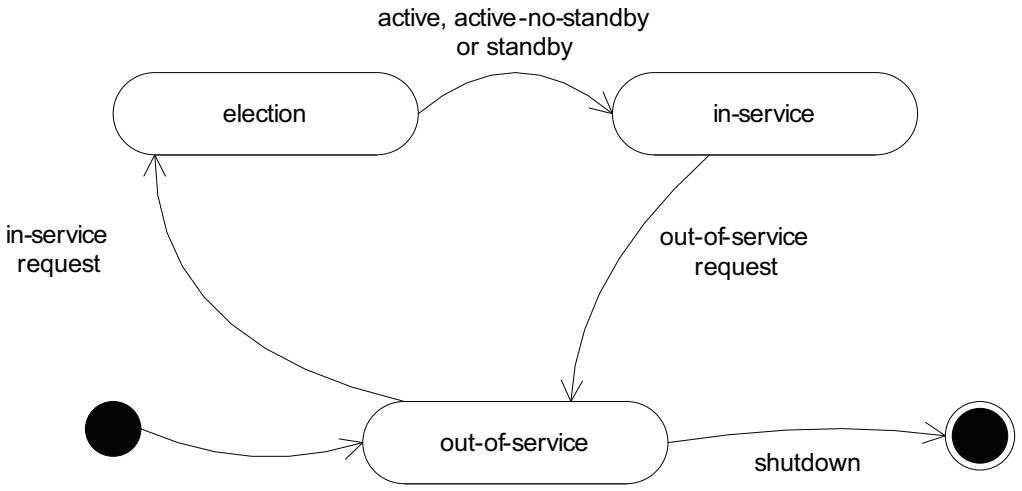
10.2 Readiness State

The RSM implements Readiness state in accordance to "Service Availability Forum Hardware Platform Interface Specification". The Readiness state indicates if an application is available to provide service. The Readiness state is defined as follows:

- Out-of-service - The RSM is up but it does not participate in chassis management. It is ready to be shut down at any point, but still operational to go to in-service state. Only a small subset of commands on the system management interface are available.
- Election - The RSM is up and runs the election process that determines the RSM's future role in chassis management (active or standby). At that moment, it does not participate in chassis management. Only a small subset of commands on the system management interface are available.
- In-service - The RSM provides service in accordance with the role determined by HA state. All commands on the system management interface are available.

Valid Readiness state transitions are presented in Figure 2.

Figure 2. Readiness State Transitions



1. The standby RSM can be taken out of service. In this case, the active RSM operates without redundancy.

The following command can be executed to set Readiness state:

```
cmmset -l cmm -d ReadinessState -v <state>
```

where *state* is one of the following:

- InService
- OutOfService

The following command can be executed to get Readiness state:

```
cmmget -l cmm -d ReadinessState
```

To get the reason for going to out-of-service, execute the command:

```
cmmget -d OutOfServiceCause
```

10.2.1 Changing Peer RSM Readiness State

To change Readiness state of the peer RSM, execute the command:

```
cmmset -l cmm -d PeerReadinessState -v <state>
```

where *state* is one of the following:

- InService
- OutOfService
- ForcedExit

The ForcedExit option causes a peer RSM process to abruptly terminate. This option may be used when a peer does not respond to other management requests.

An example scenario of a command execution in a redundant configuration is when RSM1 is active while RSM2 is standby and unresponsive. Issuing the command `cmmset -l cmm -d PeerReadinessState -v forcedexit`, RSM1 becomes active-no-standby while the RSM process on RSM2 is stopped. Next, PMS restarts the RSM process on RSM2 and RSM2 enters election state. As a result of the election process, RSM1 becomes active again while RSM2 is promoted to standby.

10.2.2 HA Redundancy Sensor

The "HA Redundancy" sensor tracks the progress of the redundancy protocol executed by RSMs. For detailed description refer to Appendix D, “OEM Sensor Events”.

10.3 HA State

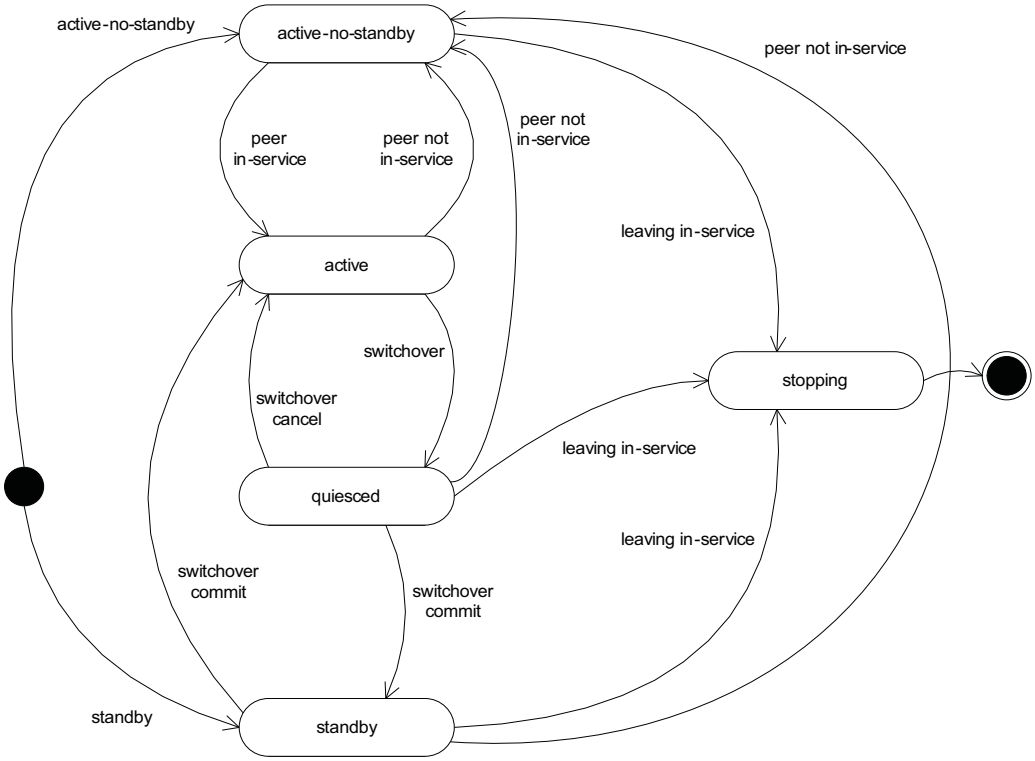
The RSM implements HA states in accordance with the “Service Availability Forum Hardware Platform Interface Specification”. The HA state indicates the role of an application in a redundant configuration while being in in-service Readiness state. The HA state is defined as follows:

- Active - The RSM executes chassis management and there is a standby RSM in the chassis. The active RSM updates the standby RSM with critical data and files.
- Active-no-standby - The RSM executes chassis management but there is no standby RSM in the chassis to communicate with. Hence, data synchronization does not occur.
- Quiesced - The RSM prepares for switchover from active RSM to standby RSM.
- Standby - The RSM accepts state updates from the active RSM.
- Stopping - The RSM no longer acts as an active or standby RSM and prepares to enter out-of-service Readiness state. All tasks in progress are being completed. The state is persisted on non-volatile storage.
- NotInService - The RSM is not in its in-service Readiness state.

Note: From the user interface point of view, the Active and Active-no-standby states are almost the same. They accept the same CLI commands except for commands related to switchover. For the sake of simplicity, this document uses the term “active RSM” to describe an RSM in one of these two HA states as long as no ambiguity arises.

Valid HA state transitions are presented in Figure 3.

Figure 3. High Availability State Transitions



The following command can be executed to get the HA state:

```
cmmget -l cmm -d HaState
```

10.3.1 Presence State

In addition to the above, an RSM is always in one of these presence states: - present or absent.

The following command can be executed to get the presence, Readiness, and HA states of RSMs:

```
cmmget -l cmm –d redundancy
```

This command also displays which RSM you are currently logged in to. When you are looking at the front of a chassis, the RSM on the left is designated as RSM1 and the RSM on the right is designated as RSM2.

10.3.2 HA State Sensor

The “HA state” Sensor tracks Readiness and HA states assumed by the RSM. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.3.3 In-service Request Sensor

The “In-service Request” sensor indicates the reason for transitioning to in-service. This is a SEL type sensor that makes a SEL entry but cannot be queried through the system management interface. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.3.4 Out-of-service Request Sensor

The “Out-of-service Request” sensor indicates the reason for transitioning to out-of-service. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.3.5 Redundancy Sensor

The “Redundancy” Sensor tracks HA election and connection setup progress. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.4 Health Score

The health of the RSM is determined by computing its health score. The health score is presented as an ordered sequence of three scores, one for each severity:

<critical_score major_score minor_score>

The score for a severity is calculated as:

*<severity>_score = round(255 * current / maximum)*

The current value is the sum of weights for sensors contributing to the RSM’s health that have asserted health events for this severity. The maximum value is the sum of weights for all sensors contributing to the RSM’s health for this severity. The score is normalized to range <0,255>.

The health score is an inverted indicator of the RSM’s health: the lower health score means better health. To retrieve the current health score, execute the CLI command:

```
cmmget -d HaHealthScore
```

Health score comparisons are made with strict priority order between severity scores. For example:

1) RSM1:active: <0 0 10> / RSM2:standby: <0 20 0>

2) RSM1:active has a critical event

3) RSM1:active has health score: <10 0 10>

4) RSM1 health is now worse than RSM2 health, so switchover is performed

5) RSM1:standby: <10 0 10> / RSM2:active: <0 20 0>

For the health score comparisons, an additional algorithm is used that prevents frequent switchovers.

Event contributions to health score and weights are configurable properties that are maintained in the `/etc/cmm/events.conf` file. Each health event has a default weight of one assigned to it, causing all health events to have equal importance in affecting health score.

10.4.1 Health Score Sensor

The “Health Score” Sensor logs changes to the health score value. This is an event-only sensor. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.5 Data Synchronization

To ensure that critical data on the standby RSM matches the data on the active RSM, the active RSM synchronizes the data and configuration files on the standby RSM with its own data and configuration files. The RSM uses an SCTP connection between Active and Standby as the data transport layer for data synchronization.

For synchronization to occur, both of the following must be true.

- The two RSMs must be able to communicate with each other over their dedicated IPMB connection. This is required for LISM IP addresses exchanged during election.
- The two RSMs must be able to communicate with each other over an Ethernet connection. All data items and files will be synchronized over this connection.

  The two RSMs can have an Ethernet connection through the Ethernet switches in the chassis, which requires that both switches be present. The RSMs can also have a connection through an external Ethernet switch connected to either the front or the rear ports. Lastly, they can have a connection using a crossover cable connecting the two front ports of the RSMs.

The only data “synchronized” between RSMs over IPMB are the IP addresses of each RSM so the synchronization process can establish a connection over the Ethernet. Once the connection is in place, all data and files are synchronized over the Ethernet.

There are two types of data synchronization: initial synchronization and partial synchronization.

The RSMs initially synchronize data and files from the active to the standby RSM just after booting the RSM firmware. Inserting a new RSM into the chassis also causes a full synchronization from the active RSM to the newly inserted standby RSM.

When the active RSM synchronizes configuration files between the two RSMs, the active RSM overwrites all the existing files on the standby RSM with files from the active RSM.

As far as critical data is concerned, partial synchronization occurs automatically whenever some critical data item on the active RSM changes.

Files are only synchronized upon changes caused by user actions on system management interfaces. Manual changes or touching with the Linux* `touch` command have no direct effect on file synchronization.

Some special cases of synchronization are described in the following sections.

Table 14 lists the items that are synchronized between the active and the standby RSMs. During a full synchronization all of these files and data are synchronized. A change to any one of these files or data items causes synchronization.

Table 14. RSM Synchronization Files and Data (Sheet 1 of 2)

File(s) or Data	Description
IP Address Settings	Current IP address settings for the eth0, eth1, eth2, eth3, and eth1:1 ports
Ekey Controller Structures	Ekey Controller Structures
Bused EKey States	Bused EKey States
Fan States	Fan States
Cooling State	Cooling State information
SDR structures	SDR structures
Hot Swap FRU state, Power Usage and Power Info	Hot Swap FRU state, Power Usage and Power Info
FIM FRU Caches	FIM FRU Caches
SEL Events	Individual SEL Events
/var/log/cmm/sel/sel.dat	System Event Log

Table 14. RSM Synchronization Files and Data (Sheet 2 of 2)

File(s) or Data	Description
/etc/cmm/*.conf	RSM configuration files (except for pm.conf, events.conf, local.conf)
/etc/passwd	Password file
/etc/shadow	Password file
/etc/group	Group file
/usr/share/cmm/scripts	User scripts directory

10.5.1 Time and Date Synchronization

RSMs perform continuous time and date synchronization using the NTP (RFC-1305) client-server synchronization model. Within this model, the active RSM acts as an NTP Server, providing reference time, while the standby RSM acts as an NTP Client synchronizing its internal time to that provided by the NTP Server. Time and date synchronization is managed by a separate process (ntpd), and is an independent mechanism from the one used for synchronization of other data. The NTP time synchronization model provides for better stability of the calendar time compared to the one used in prior firmware versions, but it reacts with inertia to discontinuous time changes induced by the operator using the `date` command.

See Section 29.0, “Time Synchronization” on page 148 for more details on NTP and time synchronization in the RSM.

10.5.2 User Scripts Synchronization

User scripts located in directory `/usr/share/cmm/scripts` are synchronized after RSMs establish communication. In addition, a particular script is synchronized when a new event-to-script association is made for this script.

Other than that, user scripts are not subject to partial synchronization *unless* it is specifically requested it using a CLI command after applying editorial changes to the script. To force synchronization of a particular script after an editorial change, execute the command:

```
cmmset -l cmm -d synchronizescript -v <scriptname>
```

The configuration parameter `SyncUserScripts` stored in the RSM configuration file `/etc/cmm/shm.conf` controls synchronization of user scripts between RSMs running different versions of the firmware. If the firmware versions on the two RSMs are the same, this flag is ignored.

You can query the current value of this parameter using the CLI command `cmmget` and set it to the desired value using the CLI command `cmmset`. These commands can also be executed using the SNMP and ShM API interfaces.

To set the value of the scripts synchronization flag, execute this command:

```
cmmset -l cmm -d syncuserscripts -v <syncflag>
```

In version 8.x, the following value can be assigned to `<syncflag>`:

`always` — Synchronizes user scripts no matter what firmware version the other RSM is running.

To query the value of the script synchronization flag, execute this command:

```
cmmget -l cmm -d syncuserscripts
```

The returned value is `always`.User scripts are always synchronized between the RSMs.

See Chapter 20.0, “RSM Scripting” on page 103 for more details on RSM scripting feature.

10.5.3 Data Synchronization Failure

If an active RSM encounters a failure during the data synchronization process, it stops synchronization and goes to active-no-standby state. The standby RSM transits to out-of-service state, sets the cause of transition on the “Out-of-service Request” sensor, logs a SEL event, and sends an SNMP trap. Next, it goes back to election state, where it tries to reconnect to the active RSM. As soon as the RSM completes the election process and regains standby state, initial synchronization begins.

10.5.4 Heterogeneous Synchronization

RSM version 8.x is not backward compatible with prior firmware versions in terms of data synchronization. However, RSM version 8.x supports heterogeneous synchronization with higher firmware versions.

10.5.5 DataSync Status Sensor

The “DataSync Status” sensor tracks the data synchronization status. RSM version 8.x does not classify the synchronized data as priority 1 and priority 2. This sensor can only be queried through the active RSM.

For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.5.5.1 Sensor bitmap

The "DataSync Status" sensor is a discrete Radisys OEM sensor with status bits representing the state of different parts of the Data Synchronization module:

Bit 0 (Running) is set when the Data Synchronization module is active.

Bit 1 (P1Done) is set when all Priority 1 data have been synchronized between the two RSMs. This bit is cleared when there is Priority 1 data that needs to be synchronized.

Bit 2 (P2Done) is set when all Priority 2 data have been synchronized between the two RSMs. This bit is cleared when there is Priority 2 data that needs to be synchronized.

Bit 3 (InitSyncDone) is set when both Priority 1 and Priority 2 data have been synchronized. This bit stays set (latches) until the RSM changes between active and standby or loses contact with the other RSM.

Note: When data synchronization starts for the first time and whenever an RSM changes between active and standby, the status bits in the DataSync Status sensor are all reset to 0x0000.

10.5.5.2 Querying the DataSync Status sensor

The status of the DataSync Status sensor can be queried using the following CLI command:

```
cmmget –l cmm –t "0:DataSync Status" –d current
```

Note: This command can be executed only on the active RSM.

Output of the command is as follows:

Initial state; single RSM in the chassis:

```
The current value is 0x0000

DataSync disabled - there is no partner CMM present
```

Initial data synchronization in progress:

```
The current value is 0x0001

Initial Data Synchronization not complete

   There is Priority 1 data to sync

   There is Priority 2 data to sync

No Data Synchronization problems known
```

Initial data synchronization is complete:

```
The current value is 0x000f

Initial Data Synchronization complete

   Priority 1 Data is synced

   Priority 2 Data is synced

No Data Synchronization problems known
```

10.6 Failover and Switchover

Once data has been synchronized between the two RSMs, the active RSM constantly monitors its own health as well as the health of the standby RSM. In the event of one of the scenarios listed in the sections that follow, the active RSM hands over control to the standby RSM. In accordance with the Service Availability Forum redundancy model, two distinct methods are used:

- switchover
- failover

10.6.1 Switchover

Switchover is a graceful transfer of control from the active RSM to the standby RSM. As a result of switchover, the standby RSM becomes active and the active RSM becomes standby. The following preconditions must exist before switchover can take place:

- There are redundant RSMs in the chassis assigned with active/standby states
- RSMs can communicate over IPMB and Ethernet
- RSMs are synchronized

These are the switchover procedure types:

- automatic switchover
- manual switchover
- legacy switchover

10.6.1.1 Automatic Switchover

Automatic switchover is caused by health degradation of the active RSM. Automatic switchover is possible in automatic switchover mode, which is the default mode of the RSM's operation.

While in automatic switchover mode, the active RSM periodically monitors the health of the standby RSM. When the active RSM sees that it has become less healthy than the standby RSM, it proposes switchover. The standby RSM may reject this proposal if its health has degraded recently. If the standby RSM accepts the proposal, switchover occurs.

10.6.1.2 Manual Switchover

Manual switchover is user-requested through the system management interface or is a part of the in-service exit procedure. This switchover is forcible: the standby RSM cannot reject it.

The following CLI command triggers manual switchover:

```
cmmset -l cmm -d switchover -v manual
```

A manual switchover using the command above can be initiated only on the active RSM.

The other possible reasons for manual switchover are as follows:

- the ejector latch on the active RSM is opened
- the active RSM is rebooted

When manual switchover occurs, the standby and active RSMs switch their HA states. The new active RSM enters manual switchover mode and does not start to monitor the standby RSM's health until one of the following happens:

- the automatic switchover command is issued on the active RSM:
  ```
  cmmset -l cmm -d switchover -v automatic
  ```
- the active RSM leaves active HA state

As a result, the RSM is placed back in automatic switchover mode. A user-triggered return to automatic switchover mode after manual switchover ensures that user selection as to which RSM is the active one is not overridden.

10.6.1.3 Remote Manual Switchover

You may also request manual switchover from the standby RSM. To initiate remote manual switchover, execute the command:

```
cmmset -l cmm -d PeerSwitchover -v manual
```

When the active RSM receives a switchover request from the standby RSM, it executes the procedure described in Chapter 10.0, "Manual Switchover" on page 57.

10.6.1.4 Legacy Switchover

The following legacy command can be issued to the active RSM to switchover to the standby RSM:

```
cmmset -l cmm -d failover -v <mode>
```

The argument <mode> to the `-v` parameter is one of the following:

- 1 — Switchover to the standby RSM only if it is running the same version of the firmware as the active RSM or a later version of the firmware.
- any — Switchover to the standby RSM regardless of the version of the firmware that the standby RSM is running.

When this command is completed, both the active and standby RSMs remain in automatic switchover mode. A health change may cause a switchover.

A legacy switchover using the command above can be initiated only on the active RSM.

10.6.2 Failover

Failover is the ungraceful transfer of control to the standby RSM due to failure of the active RSM. Failover does not guarantee that all critical data from the active RSM is synchronized to the standby RSM.

The following scenarios cause a failover as long as the standby RSM is operational, even when it is not as healthy as the active RSM:

- Loss of IPMB connectivity
- The HEALTHY# hardware signal for the active RSM is asserted
- The active RSM is abruptly removed from the chassis

10.6.3 Standby Reboot

To reboot the standby RSM from the active RSM, execute the command:

```
cmmset -d StandbyCmmReboot -v 1
```

10.6.4 HA Control Sensor

The RSM supports the “HA control” Sensor. This sensor logs events related to HA control events and commands. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

10.7 CMM Status Sensor

The RSM supports the “CMM Status” Sensor. The “CMM Status” sensor events announce when the RSM firmware is or is not fully up and running and ready to process all requests.

The “CMM Status Ready” event is deasserted on the active RSM while it is powering up. It is also deasserted on the standby RSM after it transitions to active mode during a failover. The event is asserted only on the active RSM. The “CMM Status Ready” event is asserted after the RSM firmware is fully initialized and operational. The major difference to prior firmware versions is that the running bit is used for Readiness and HA state indications. For a detailed sensor description, refer to Appendix D, “OEM Sensor Events”.

Chapter 11

11.0 Re-enumeration

11.1 Overview

Re-enumeration provides a way to recover from situations such as double failures (both RSMs have failed or have been removed from the chassis). Re-enumeration is also performed after chassis power up and after failover. The RSM first determines whether or not it is the active RSM. The standby RSM does not re-enumerate; instead, it relies on the information synchronized from the active RSM. The active RSM performs the process of re-enumeration to discover the information it needs about the devices in the chassis. Re-enumeration does not involve restarting the individual blades present in the chassis. After startup the active RSM determines the entities present in the chassis. Thereafter, the RSM queries each present entity to get state and other information. The RSM re-enumeration process obtains the following information for each FRU in the chassis:

- Presence
- Hot Swap State
- Power Usage
- Sensor Data Records
- Platform Events
- Board EKey Usage
- Bused EKey Usage

11.2 Re-enumeration Sensor

The “Re-enumeration State” Sensor tracks the progress of the re-enumeration process. For a detailed description, refer to Appendix D, “OEM Sensor Events”.

11.3 Event Regeneration

During the re-enumeration process, the RSM sends out the “Set Event Receiver” command to all the entities in the chassis. On receiving the command, the entities re-arm event generation for all their internal sensors. This causes them to transmit the event messages that they currently have based on existing event conditions. These events are logged in the SEL.

The regeneration of events may cause events to be logged into the SEL twice. This double logging will cause user scripts associated with those events to run twice.

11.4 Cooling

If the RSM detects a fantray during re-enumeration, it automatically sets the fan speeds to the maximum level. The speeds are not brought back to normal level until re-enumeration is finished and the RSM has determined that there are no thermal events in the chassis.

11.5 Resolution of EKeys

During re-enumeration the RSM determines the status of EKeys for the boards present in the chassis. If there are interfaces that can be enabled with respect to the other end-point, the RSM completes the EKeying process as described in Section 24.0, “Electronic Keying Management” on page 121.

If there are EKeys enabled to a slot but the RSM cannot discover a board in that slot, the RSM assumes that the board actually is in that slot but in the M7 (Communication Lost) state. However, if there is no board in the slot, the `cmmset` command should be executed using the `fruextractionnotify` dataitem so the RSMs know that the slot is empty:

```
cmmset –l <location> -d fruextractionnotify –v 1
```

12.0 Process Monitoring and Integrity

12.1 Overview

The shelf manager module (RSM) monitors the general health of processes running on the RSM and can take recovery actions upon detection of failed processes. This is handled by the Process Monitoring Service (PMS).

Upon detecting unhealthy processes, the PMS will take a configurable recovery action. Examples of recovery actions include restarting the process and failing over to the standby RSM.

The PMS periodically strobes the hardware watchdog. This ensures that when the PMS fails a corrective action is automatically taken by initiating a failover and resetting the RSM.

All the configuration parameters for the PMS are stored in file `/etc/cmm/pm.conf`. This configuration file is read only once by the PMS at the time of initialization. If an error is encountered during parsing the configuration file, the PMS uses a default configuration as specified later in this chapter.

The PMS can monitor processes that already exist when it starts, or it can also start the processes and then monitor them. The PMS supports two types of process monitoring:

- Monitoring for existence of a process
- Monitoring for existence and integrity. Integrity monitoring is done by a separate process called Process Integrity Executable (PIE).

The configuration lets you tune the system parameters for the given platform. Examples of parameters include:

- Monitoring interval—Time between successive health checks of processes
- Number of retries—Maximum number of recovery attempts (within a specific time interval) beyond which the PMS either escalates the recovery action or stops monitoring
- Ramp-up times—Time interval after a process has been recovered that must elapse before the PMS resumes monitoring the process
- Recovery-actions—Different recovery actions to recover from a failed/unresponsive process

12.1.1 Process Existence Monitoring

Process existence monitoring checks whether a process exists by inspecting the process table for the operating system. When the RSM firmware is started, the PMS determines the set of processes it should monitor for existence. The PMS periodically queries the operating system to determine if those processes still exist. When a monitored process is found not to exist, the PMS generates an event to be logged in the SEL and then executes the recovery action defined for such an event.

Process existence monitoring can be utilized on all permanent processes (processes that exist as long as the RSM firmware is running). This is particularly useful when monitoring processes that are not part of the RSM firmware itself, such as syslog-ng and crond on the Linux* operating system or user scripts.

12.1.2 Process Watchdog Monitoring

Process watchdog monitoring requires that the process being monitored notify the PMS of its continued operation. Notifying the PMS allows the PMS to monitor the process for existence and to detect the conditions where a process has locked up. If the PMS determines that a process is not responsive (that is, the process stops notifying the PMS of its continued operation), the PMS generates a SEL entry and takes the configured recovery action.

12.1.3 Process Integrity Monitoring

Existence monitoring simply detects whether the expected process exists. If the process crashes, it will be recovered quickly. However, if the process continues to exist but is not functioning as it should (for example, it is caught in a loop), existence monitoring will not detect this. Process Integrity Monitoring offers a way to inspect the proper behavior of a monitored process through further interaction with the monitored process. A special executable called Process Integrity Executable (PIE) is used for this purpose.

A PIE is responsible for determining the health of a process or processes. A PIE runs periodically to interact with the process it is monitoring (for instance, by running a loopback command through the message queues) to determine whether it is responsive.

When a PIE finds an unhealthy process, it notifies the PMS of the errant process so that the PMS can take the appropriate action.

An example of a PIE would be one that monitored the Simple Network Management Protocol (SNMP) process. The PIE could utilize SNMP get operations to query the SNMP process. If the SNMP process cannot respond to the queries with the appropriate information, the process would be considered unhealthy and the PIE would notify the PMS.

Since PIEs can be written in many different ways, the fault conditions it can detect will vary. For example, if a PIE utilizes process commands, as described in the example above, process integrity monitoring can detect process existence, thread lock-ups, and if the process is functioning properly. If a PIE just audits the process' data it cannot necessarily detect lock-ups because the data could have been in a valid state when it locked-up. Also, depending on the particular instance, process integrity could potentially be a very intensive operation and therefore should only be done at a longer interval, such as hours.

12.2 Processes Monitored

The pm.conf file contains the full list of all processes monitored by PMS in the default configuration.

12.3 Process Monitoring Targets

Every monitored process is available as a target for the ‘cmm’ location. Use the following CLI command to view the targets for the processes being monitored:

```
cmmget -l cmm -d listtargets
```

All monitored processes appear as a target in the form of PmsProc*n* where *n* stands for the process unique ID.

The particular processes currently being monitored are listed in the output returned from the above command. The targets that pertain to process monitoring have the form PmsProc*n*, where *n* is a one-digit, two-digit, or three-digit number.

To view the name of a monitored process use the following command:

```
cmmget -l cmm -t PmsProc<N> -d processname
```

For example, the command

```
cmmget -l cmm -t PmsProc51 -d processname
```

returns this output:

```
snmpd
```

12.4 Process Dependency

The PMS can also start processes before starting to monitor them. Defining Process Dependency allows the PMS to start the monitored processes in specific order. This is achieved by using an optional parameter P*n*_STARTED_AFTER. This parameter holds the value of a unique ID for another monitored process. For example, the default PMS configuration has the following definition for snmpd monitoring defined as follows:

```
P11_STARTED_AFTER = 1
```

The above line states that the process with unique ID 11 should be started only after the process with unique ID 1 has been started. For a detailed description of parameter definitions, refer to Section 12.9.1, “Configuration Parameters” on page 72.

Note: The process dependency information is used only when the PMS initializes and starts the processes. The dependency information is ignored when restarting a process in case of a failure.

12.5 Peer Processes

PMS allows a monitored process configuration to define a peer process. When the parameter Pn_PEER_PROCESS is defined for a monitored process, it shares the recovery action and escalation action of the peer process.

For example, if the PMS configuration file contains the entry P51_PEER = 2, then the failure of either Process 51 or Process 2 causes a recovery action to be performed for both Process 51 and Process 2.

For a detailed description of parameter definitions, refer to Section 12.9.1, “Configuration Parameters” on page 72.

12.6 Process Monitoring Dataitems

Table 15 lists the dataitems used to configure (`cmmset`) and retrieve (`cmmget`) information about the Process Monitoring Service. Specify the `cmm` location (with no sub-FRU ID) and a target of `PmsProc`*n* (where *n* is a one-digit, two-digit, or three-digit number).

Table 15. Dataitems for Process Monitoring

Dataitem	Description	Get/ Set	CLI Get Output	Valid Set Values
AdminState	A target of "PmsProc[#]" gets or sets the unique state of an individual process, where # is the unique process number for the process. This dataitem is maintained separately on each RSM and is not synched between RSMs. This allows independent control of each RSM's adminis-trate. Can be set on either the active or the standby RSM.	Both	"1:Unlocked" or "2:Locked"	1 - Unlocked 2 - Locked
RecoveryAction	Used to query the recovery action of a process monitored by PMS. **Note:** Valid only for a target of "PmsProc*n*", where *n* is the unique number denoting that process.	Get	"1:No Action", "2:Process Restart", "3: Failover & Restart", or "4:Failover & Reboot"	1 - no action 2 - process restart 3 - failover & restart 4 - failover & reboot
EscalationAction	Used to query the process restart escalation action. **Note:** Valid for a target of "PmsProc*n*", where *n* is the unique number denoting that process.	Get	"1:No Action", "2:Failover & Reboot"	1 - no action 2 - failover & reboot **Note:** Setting this dataitem to "no action" is not normally recommended.
ProcessName	Used to query the process name of the monitored process. A target of "PmsProc*n*" retrieves the name of an individual process, where *n* is the unique number denoting that process.	Get	"<Process_Name>"	N/A
OpState	Used to query the operational state of a monitored process. An operational state of disabled indicates that the process has failed and cannot be recovered Valid targets are: "PmsProc*n*" where *n* is the unique number to denote that process	Get	"1:Enabled", "2:Disabled"	N/A

12.6.1 Examples

The following example gets the recovery action assigned to a monitored process:

```
cmmget -l cmm -t PmsProc51 -d RecoveryAction
```

12.7 Process Monitoring RSM Events

The "Process Monitoring Service" sensor types are used to assert and de-assert process status information such as process presence not detected, process recovery failure, or recovery action taken.

Event severities are configurable by the user and are unique to the process being monitored. Values for severity are: 1 = minor, 2 = major, 3 = critical.

The processes that are monitored and their default severities are listed below. Severities are configured (while the PMS is not running) by changing the P*n*_SEVERITY field in the configuration file, `/etc/cmm/pm.conf`, where *n* stands for a one-digit, two-digit or a three-digit number. The default configuration file is included at the end of this chapter.

12.8 Failure Scenarios and Event Processing

This section describes the process fault scenarios that are detected and handled by the PMS. It also describes the event processing that is associated with the detection and recovery mechanisms. Each scenario contains a brief description and a table that further describes the scenario.

Each table contains the following columns:

- The **Description** column describes the current action.
- The **Event** column defines the text for the event that is written to the SEL. The text in this field describes the portion of the event that contains the event-specific string. The remainder of the event text is standard for all events. In the case of the PMS, however, the target name (sensor name) is `PmsProc`*n* (where *n* is the unique identifier of the given process) instead of the name of the sensor.
- The **UID** column indicates the unique identifier for the process that causes the event. An ID of 1 indicates the monitoring service itself (global); an ID of # indicates an application process.
- The **Event Direction** column indicates if the event is asserted or de-asserted. For items that are just written to the SEL for informational purposes, the assertion state does not apply. However, it is required by the interface and therefore is set to de-assert.
- The **Severity** column lists the severity of the event. A severity of Configure indicates that the severity is configurable. The configurable severities are available in the Configuration Database.

12.8.1 No action recovery

The PMS detects a process fault. The configured recovery action is to take no action. The PMS disables monitoring of the process.

Table 16. No Action Recovery

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault determines the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "no action".	Take no action specified for recovery	#	N/A	Configure
No attempt is made to recover the process. The PMS stops monitoring the process. See Section 12.8.11, "Process administrative action" on page 71, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assertion	Configure

12.8.2 Successful restart recovery

The PMS detects a process fault. The configured recovery action is to restart the process. The PMS is able to successfully recover the process by restarting it.

Table 17. Successful Restart Recovery

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault determines the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "process restart".	Attempting process restart recovery action	#	N/A	Configure
PMS was successfully able to restart the process	Recovery successful	#	Deassertion	OK

12.8.3 Successful failover and restart recovery

The PMS detects a process fault. The configured recovery action is to failover to the standby RSM and then restart the failed process. The PMS is able to successfully recover the process by restarting it.

Table 18. Successful Failover and Restart Recovery

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "failover and restart".	Attempting process failover and restart recovery action	#	N/A	Configure
PMS executes a failover. **Note:** This step is skipped when running on the standby RSM.	Failover	N/A	N/A	N/A
PMS was successfully able to restart the process **Note:** PMS executes this step even if the failover was unsuccessful (standby not available, unhealthy, and so on).	Recovery successful	#	Deassertion	OK

12.8.4 Successful failover and reboot recovery

The PMS detects a process fault. The configured recovery action is to fail over to the standby RSM, then reboot the new standby RSM once failover is complete. The PMS is able to successfully recover the process by restarting it.

Table 19. Successful Failover and Reboot Recovery

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "failover and reboot"	Attempting failover and reboot recovery action	#	N/A	Configure
PMS executes a failover. **Note:** This step is skipped when running on the standby RSM.	Failover	N/A	N/A	N/A
PMS is running on the standby RSM (failover was successful or already running on the standby). PMS recovers the RSM by rebooting. Upon initialization of PMS after the reboot the monitor desserts the event.	Monitoring initialized	#	Deassertion	OK

12.8.5 Failed failover and reboot recovery for a non-critical process

The PMS is running on the active RSM and detects a monitored process fault. The severity of the process is configured to a value that is not critical. The configured recovery action is to fail over to the standby RSM and reboot the new standby RSM. The failover recovery action is unsuccessful (standby RSM is not available, for example). The process being monitored is not of a critical severity and therefore the reboot of the RSM will not be performed.

Table 20. Failed Failover and Reboot Recovery for a Non-Critical Process

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "failover and reboot"	Attempting failover and reboot recovery action	#	N/A	Configure
PMS executes a failover	Failover	N/A	N/A	N/A
PMS detects that it is still running on the active RSM. The process is not critical and therefore the reboot operation will not be performed.	Failover and reboot recovery failure	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 12.8.11, "Process administrative action" on page 71, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assertion	Configure

12.8.6 Failed failover and reboot recovery for a critical process

The PMS is running on the active RSM and detects a monitored process fault. The severity of the process is configured to be critical. The configured recovery action is to failover to the standby RSM, then reboot the new standby RSM. The failover recovery action is unsuccessful (standby is not available, for example). The process being monitored is of a critical severity and therefore the reboot of the RSM is performed.

Table 21. Failed Failover and Reboot Recovery for a Critical Process

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "failover and reboot".	Attempting failover and reboot recovery action	#	N/A	Configure
PMS executes a failover.	Failover	N/A	N/A	N/A
PMS detects that it is still running on the active RSM. The process is critical and therefore the reboot operation is performed. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	PMS initiates a reboot; monitoring initialized	#	Deassertion	OK

12.8.7 Excessive restarts and escalation is no action

The PMS detects a process fault. The configured recovery action is to restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS executes the escalation action, which is configured for no action.

Table 22. Excessive Restarts, Escalation No Action (Sheet 1 of 2)

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "process restart"	Attempting process restart recovery action	#	N/A	Configure

Table 22. Excessive Restarts, Escalation No Action (Sheet 2 of 2)

Description	Event	UID	Event Direction	Severity
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
PMS attempts to execute the escalated recovery action. Since the recovery action is "no action", PMS disables monitoring of the process.	Take no action specified for escalated recovery	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 12.8.11, "Process administrative action" on page 71, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assertion	Configure

12.8.8 Excessive restarts and successful failover/reboot escalation

The PMS detects a process fault. The configured recovery action is to restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS executes the escalation action. The configured escalation recovery action is to fail over to the standby RSM, then reboot the new standby RSM. The escalated recovery action is successful.

Table 23. Excessive Restarts, Successful Escalation of Failover and Reboot

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover and reboot escalated recovery action	#	N/A	Configure
PMS executes a failover. **Note:** This step is skipped when running on the standby RSM.	Failover	N/A	N/A	N/A
PMS is running on the standby RSM (failover was successful or already running on the standby), PMS recovers the RSM by rebooting. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	Monitoring initialized	#	Deassertion	OK

12.8.9 Excessive restarts, failed failover / reboot escalation, non-critical process

The PMS detects a process fault. The severity of the process is configured to a value that is not critical. The configured recovery action is to restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS executes the escalation action. The configured escalation recovery action is to fail over to the standby RSM, then reboot the new standby RSM. The failover recovery action is unsuccessful (standby is not available, for example). The process being monitored is not of a critical severity. Therefore, the RSM is not rebooted.

Table 24. Excessive Restarts, Failed Escalation of Failover and Reboot, Non-Critical Process

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine the type of event.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover and reboot escalated recovery action	#	N/A	Configure
PMS executes a failover.	Failover	N/A	N/A	N/A
PMS detects that it is still running on the active RSM. The process is not critical and therefore the reboot operation will not be performed.	Failover and reboot escalated recovery failure	#	N/A	Configure
No attempt will be made to recover the process. The PMS will stop monitoring the process. See Section 12.8.11, “Process administrative action” on page 71, for information about how to re-enable monitoring and de-assert the event.	Process existence fault; monitoring disabled or Thread watchdog fault; monitoring disabled or Process integrity fault; monitoring disabled	#	Assertion	Configure

12.8.10 Excessive restarts, failed failover / reboot escalation, critical process

The PMS detects a process fault. The severity of the process is configured as critical. The configured recovery action is to restart the process. However, the PMS also detects that the process has exceeded the threshold for excessive process restarts. Therefore, the PMS executes the escalation recovery action. The configured escalation recovery action is to fail over to the standby RSM, then reboot the new standby RSM. The failover recovery action is unsuccessful (standby is not available, for example). The process being monitored is of critical severity. Therefore, the RSM is rebooted even though it is still the active RSM.

If the PMS detects that the process has exceeded the threshold for excessive process reboots (3 times in 900 sec), the PMS Fault sensor triggers the event "Excessive reboots/failovers; all process monitoring disabled". Reboots are then stopped, corrective action must be taken, and the RSM must be manually rebooted.

Table 25. Excessive Restarts, Failed Escalation Failover and Reboot, Critical Process

Description	Event	UID	Event Direction	Severity
PMS detects a faulty process. The mechanism (existence, thread watchdog, or integrity) used to detect the fault will determine which of the event type strings will be used.	Process existence fault; attempting recovery or Thread watchdog fault; attempting recovery or Process integrity fault; attempting recovery	#	Assertion	Configure
The recovery action specified is "restart process"	Attempting process restart recovery action	#	N/A	Configure
PMS detects that the process has been restarted excessively.	Recovery failure due to excessive restarts	#	N/A	Configure
The escalated recovery action specified is "failover and reboot"	Attempting failover and reboot escalated recovery action	#	N/A	Configure
PMS executes a failover.	Failover	N/A	N/A	N/A
PMS detects that it is still running on the active RSM. The process is critical and therefore the reboot operation is performed. Upon initialization of PMS after the reboot. The monitor will de-assert the event.	PMS initiates a reboot; monitoring initialized	#	Deassertion	OK

12.8.11 Process administrative action

The PMS has detected a fault in a process, but has not been able to recover the process (recovery is configured for no action, for example). This causes the PMS to operationally disable monitoring of the process. To re-enable monitoring of the process, an operator must administratively lock the process, take the necessary actions to fix the process, then administratively unlock the process.

Table 26. Administrative Action

Description	Event	UID	Event Direction	Severity
Operator administratively locks monitoring of the process	N/A	N/A	N/A	N/A
Operator fixes the problem	N/A	N/A	N/A	N/A
Operator administratively unlocks monitoring of the process which restarts monitoring	Monitoring initialized	#	Deassertion	OK

12.9 Configuration

The `/etc/cmm/pm.conf` file is the configuration file for the Process Monitoring Service (PMS) and Process Integrity Executable (PIE). It contains all of the non-volatile configuration data for the PMS and the PIE. It is an ASCII file that can be edited with any text editor. '#' is treated as a comment character. All text after '#' until the end of the line is treated as a comment. Blank lines are ignored.

Note: Any changes made to the `pm.conf` file will be overwritten updating the RSM firmware. Save the `pm.conf` file to a storage device or location off of the RSM before updating the firmware so the file can be restored after the update.

12.9.1 Configuration Parameters

Each target process to be monitored needs to have certain mandatory parameters defined in the `pm.conf` file. A unique ID is assigned to each monitored process. All parameters names associated with a process will have a prefix of the form P*n*_ where *n* can be any number in the range of 2-255 representing the unique ID assigned to the monitored process, e.g. P2_MONITORED_NAME, P2_MONITORING_TYPE and so on. For example, the severity parameter for a monitored process with unique ID 13 will be defined like:

```
P13_SEVERITY = 1
```

Note: The ID 0 is reserved. The ID 1 is reserved for the Process Monitoring Service itself.

12.9.1.1 P*n*_MONITORED_NAME

Defines the process name as it appears in the `/proc/[OS PID]/stat` file. OS PID refers to the Process ID.

Values: N/A.
Default: None.

12.9.1.2 P*n*_MONITORING_TYPE

This parameter determines the monitoring type. The default method is to monitor the process termination signal. The option is that a process proactively notifies its presence. The presence notification can be done in two ways, by a UDP message or a PM API call. This parameter is optional. When not specified, the monitoring type will have the default value.

Values: 1 = OS signal, 2 = OS signal and UDP message, 3 = OS signal and PM API call.
Default: 1.

12.9.1.3 P*n*_RAMP_UP_TIME

The amount of time in seconds necessary for the process to initialize and be functional. This parameter is valid only in case the monitoring type has the value: 2 or 3. In case a process does not report to PMS its continued operation within the time, the process triggers a watchdog fault. This parameter is optional.

When not specified, the parameter will have the default value.

Values: 0-255.
Default: 60.

12.9.1.4 P*n*_RETRY_TIME

The amount of time in seconds that is granted to a process after is misses its report time. This parameter is valid only in case the monitoring type has the value 2 or 3. This parameter is optional. When not specified, the parameter will have the default value.

Values: 0-255
Default: 10

12.9.1.5 P*n*_GRACE_TIME

The amount of time in seconds that is granted to a process to terminate gracefully. After the grace time, the process will be terminated with a SIGKILL signal. This parameter is optional. When not specified, the parameter will have the default value.

Values: 0-255.
Default: 30.

12.9.1.6 P*n*_STARTED

Process Started by Process Monitoring. A process is started and stopped by the PM. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 = false, 2 = true
Default: 1.

12.9.1.7 P*n*_STARTED_AFTER

When specified, a process will be started during system startup after a process of the provided ID. This parameter is optional. When specified, the process must be started by the PM.

Values: process ID.
Default: 0 (a does not depend on other processes).

Note: This parameter allows establishing a dependency tree for starting a process in a specific order. Cyclic dependencies are not supported. A parsing error will occur in case of cyclic dependency and PMS will fall back on the default configuration.

12.9.1.8 P*n*_START_COMMAND

This is the command used to start the process. The process is started in two cases. The first case is when the process was started by Process Monitoring. The second case is the process is restarted during a recovery procedure and the restart command is not specified. This parameter is optional. It must be provided when a process is started by Process Monitoring or the recovery action requires a restart and there is no restart command specified.

Values: N/A.
Default: None.

12.9.1.9 P*n*_RESTART_TYPE

The type of procedure used to restart a process, in case the recovery action mandates so. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 = start/stop, 2 = restart.
Default: 1.

12.9.1.10 P*n*_STOP_TYPE

This parameter specifies the way a process is stopped. The process is stopped in two cases. The first case is when Process Monitoring is stopped and the process was started by Process Monitoring. The second case is the process is restarted during a recovery procedure and the restart command is not specified. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 – SIGTERM/SIGKILL 2 – user defined signal, 3 – stop command.
Default: 1.

12.9.1.11 P*n*_STOP_SIGNAL

This is the user defined signal used to stop a process. This parameter is optional. It must be provided when the stop type value is 2 – a user-defined signal.

Values: N/A.
Default: None.

12.9.1.12 P*n*_STOP_COMMAND

This is the command used to stop a process. This parameter is optional. It must be provided when the stop type value is 3 – a stop command.

Values: N/A.
Default: None.

12.9.1.13 P*n*_RESTART_COMMAND

This is the command used to restart a process. The parameter is optional. When specified, the command is used to perform recovery action requiring process restart. When not specified, the process stop/start command sequence is used to perform a recovery action requiring process restart.

Values: N/A.
Default: None.

12.9.1.14 P*n*_SEVERITY

An indicator for the importance of a given process. This severity will determine at what level SEL entries are generated and when reboots should occur on an active RSM. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 = minor, 2 = major, 3 = critical.
Default: 1.

12.9.1.15 P*n*_RECOVERY_ACTION

This is the recovery action to take upon detection of a failed process. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 = no Action, 2 = process restart, 3 = switchover and process restart, 4 = switchover and reboot.
Default: 1.

12.9.1.16 P*n*_RECOVERY_ESCALATION

This determines the action to take if the recovery action includes "process restart" and it fails. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1= no action, 2 = switchover and reboot.
Default: 1.

12.9.1.17 P*n*_PEER

This parameter specifies the peer process ID. This parameter is optional. When specified, the recovery action and escalation action parameters are copied from the peer process. When not specified, there is no peer for this process.

Values: N/A.
Default: None.

Note: If Pn_PEER is defined for a process, recovery and escalation parameter values defined for this process will be ignored and the values from the peer process will be used. A cyclic dependency between different monitored processes will result in a parsing error.

12.9.1.18 P*n*_ESCALATION_NUMBER

This is the number of process restarts that are allowed (within the interval specified below) before escalation starts. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 - 255.
Default: 5.

12.9.1.19 P*n*_ESCALATION_INTERVAL

Time interval in seconds during which if the number of restarts exceed the P*n*_ESCALATION_NUMBER, escalation action will be initiated for a monitored process. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 - 65535.
Default: 900.

12.9.1.20 P*n*_INTEGRITY_CHECK

Indicates if an integrity check shall be performed for a given process. This parameter is optional. When not specified, the parameter will have the default value.

Values: 1 = no integrity check, 2 = integrity check not performed.
Default: 1.

12.9.1.21 P*n*_MONITORED_NAME

This parameter is mandatory when P*n*_INTEGRITY_CHECK is set to 1. It is the process name as it appears in the /proc/[OS PID]/stat file.

Values: N/A
Default: None.

12.9.1.22 P*n*_INTEGRITY_START_COMMAND

This parameter is mandatory when P*n*_INTEGRITY_CHECK is set to 1. This is the program name and arguments used to start PIE. This parameter must be provided when the PM performs an integrity check for a given process.

Values: N/A.
Default: None.

12.9.1.23 P*n*_INTEGRITY_INTERVAL

Interval in seconds at which the integrity check probe will be started. This parameter should be provided only when P*n*_INTEGRITY_CHECK is set to 1.

Values: 1 – 65535
Default: 3600.

12.9.1.24 P*n*_INTEGRITY_REPORT_INTERVAL

This is the interval in seconds after which the probe is expected to report the integrity check result. This parameter should be provided only when P*n*_INTEGRITY_CHECK is set to 1.

Values: 1 - 255
Default: 60.

Chapter 13

13.0 Security

13.1 Role-based Access Control

RSM access control is based on the IPMI model. In this model, each user is assigned one role (privilege level). Usage of each ShM and OAM API function or IPMI command is enabled for a subset of roles. A function caller is allowed to execute the function if his role is enabled for this function. The supported roles are:

- User - Only 'benign' function calls are allowed. These are primarily commands that read data structures and retrieve status.
- Operator - All function calls are allowed, except for configuration functions that can change the behavior of the System Management interfaces. Also upgrade and downgrade initiation commands defined in ShM and OAM interface are not allowed at this level.
- Administrator - All function calls are allowed. In particular, only the user with Administrator role can manage user accounts.
- OEM - The set of function calls allowed for this role is configurable by the user.

Access control solution for ShM and OAM API is described in Section 15.3, “ShM API Access Permissions” on page 79. Access control solution for IPMI is described in Section 18.7, “RMCP Security” on page 95.

13.2 User Management

User accounts on the RSM are manageable with CLI commands. The following CLI command is used to create a user account:

```
cmmset -t User:<user_id> -d Create -v <username>:<role>:<password>
```

where:

- <user_id> is an IPMI user ID, a decimal number in the range <2, 63>. Value 2 is reserved for user `root`
- <username> is the name of the user
- <role> is a valid IPMI role assigned to the user: `user`, `operator`, `admin`, or `oem`
- <password> is the user password

RSM enforces a strong user password policy. The strong password policy is configurable using a set of configuration parameters stored in the `local.conf` configuration file.

Caution: The `local.conf` file is not replicated to the other RSM blade. Any changes to this file must be made on both RSMs.

With default strong password policy active, the newly created password must conform to the following composition rules:

- at least 8 characters in length
- at least 2 alphabetic characters
- at least 1 numeric or special character
- new password shall differ from the old password by at least 3 characters

The following CLI command is used to re-assign the user name:

```
cmmset -t User:<user_id> -d UserName -v <username>
```

The following CLI command is used to re-assign the user password:

```
cmmset -t User:<user_id> -d Password -v <passwd>
```

The new password must adhere to password composition rules listed earlier in this section.

The following CLI command is used to re-assign the user role:

```
cmmset -t User:<user_id> -d Role -v <role>
```

The following CLI command is used to retrieve the user configuration:

```
cmmget -l cmm -t User:<user_id> -d Show
```

The following CLI command is used to remove the user account:

```
cmmset -t User:<user_id> -d Delete -v 1
```

13.3 Security Sensor

The “Security” sensor is used to track security events (e.g. authentication failures detected in management layer interfaces). For a detailed description, refer to Appendix D, “OEM Sensor Events”.

Chapter 14

14.0 Hardware Platform Interface

14.1 Overview

The RSM supports Hardware Platform Interface version B.01.01. The HPI is an industry standard interface defined by Service Availability Forum to monitor and control highly available systems. The HPI allows user applications and middleware to access and manage hardware components via a standardized interface.

Detailed specification of HPI can be found in "Service Availability Forum Hardware Platform Interface Specification".

14.2 OpenHPI*

To use HPI, the System Management application must be linked with the OpenHPI* library. OpenHPI* library is an open source implementation of HPI that is compliant with version B.01.01. The OpenHPI* library has two major parts, the core library (infrastructure), and the plug-ins. The core OpenHPI* library is a dynamic library, written in the C language. The plug-in mechanism allows OpenHPI to support numerous hardware types without requiring any core changes to the library.

The OpenHPI* core library is not provided as part of the RSM firmware release. It is open source software and official support for it is not provided by Radisys.

More details about the OpenHPI* project can be found at http://www.openhpi.org/.

14.3 RSM Plug-in to OpenHPI*

Radisys provides an RSM plug-in to the Open HPI* library. The RSM plug-in provides support for calling remotely HPI interface functions on the active RSM. The plug-in implements the ATCA-to-HPI mapping as defined by "Service Availability Forum Hardware Platform Interface Specification". The plug-in communicates with the remote RSM using the Remote Shelf Management and OAM API library.

The RSM plug-in to the Open HPI* library is a part of the RSM firmware distribution. An installation guide is included in the README file located in the /src directory of the release package.

The RSM plug-in is resilient to RSM failovers. It monitors the status of the HPI connection with the remote RSM. When a connection fails, the plug-in reestablishes the connection and performs audit procedure to ensure that it presents a coherent view of the remote system.

15.0 Shelf Management & OAM API

15.1 Overview

The RSM supports Remote Shelf Management and the OAM interface. The Shelf Management interface exposes functions that correspond to IPMI commands defined in IPMI / PICMG specifications. The OAM interface defines new functions that cover functionality not defined in IPMI/ PICMG specifications, such as firmware upgrades and diagnostics.

The System Manager application calls Shelf Management and OAM API functions locally from the client library. The calls are transported to the remote RSM using a standard RPC protocol defined in *RFC1057*. The RPC messages are transported over LAN using RMCP packets. The OEM payload mechanism defined in RMCP+ encapsulates RPC into RMCP. This transport option makes it possible to utilize security features defined in RMCP+ which are not present in the RPC protocol itself.

A detailed definition of the Shelf Management & OAM API is in the "A6K-RSM-J, MPCMM0001 and MPCMM0002 Chassis Management Module ShM & OAM API Reference Manual".

15.2 Shelf Management and OAM API Client Library

The Shelf Management and OAM API client library is a dynamic library written in the C language. The client library is linked with the System Management application, and provides support for establishing a session to the Shelf Management and OAM API Server running on RSM and invoking Shelf Management and OAM functions remotely.

15.3 ShM API Access Permissions

Each time some ShM API function is called, the RSM checks if the caller has sufficient access permissions to use this function. To do so, the RSM consults the access permissions table for the ShM API. The table contains a number of rows, one per ShM API function, whereby each row stores access permission data for operator, user, and OEM roles. The administrator permissions' values are not stored in the table because the administrator, by definition, has access to all functions. Operator and user permissions are hard-coded and not editable. In contrast, "OEM" role permissions are modifiable.

The following CLI command (all on one line) is used to modify access permissions for an "OEM" role:

```
cmmset -t Func:<pnum>:<fnum> -d OemPermission -v <0|1|disabled|enabled|
reset>
```

where pnum and fnum are RPC program and function numbers identifying ShM API function.

The following CLI command is used to get access permissions for an "OEM" role:

```
cmmget -t Func:<pnum>:<fnum> -d OemPermission
```

Permission is one of the values 0, 1, disable, enable, or reset.

The RSM defines default access permissions for the "OEM" role. Default access permissions are used whenever user selected access permissions data is missing.

The following CLI command is used to set default OEM access permission settings for ShM API functions:

```
cmmset -d DefaultOemPermission -v <permission>
```

The following CLI command is used to retrieve the default OEM access permission settings for ShM API functions:

```
cmmget -d DefaultOemPermission
```

The access permissions table is stored in file `/etc/cmm/permissions.conf`. The file is owned by root and is only writable by the owner.

Chapter 16

16.0 Command Line Interface

16.1 Overview

The Command Line Interface (CLI) of the RSM connects to and communicates with the RSM as well as the intelligent devices in the chassis. The CLI is an application that runs on top of the ShM and OAM API, and it can be accessed either from the bash shell prompt (command line) or through a higher-level management application. Using the CLI, users can access information about the current state of the system, including current sensor values, threshold settings, recent events, and overall chassis health.

The CLI functions are also available through SNMP get and set commands and through the legacy RPC (Remote Procedure Call) interface. The equivalent set of functions is exposed through the ShM & OAM API.

Administrators can access the CLI through SSH (secure shell) or a Telnet session after logging in to the RSM.

CLI syntax and arguments are defined in "Alert Standard Format (ASF) Specification version 2.0".

For a complete list of commands accessed through the CLI, see the "Command Line Interface Reference for CMMs A6K-RSM-J, MPCMM0001, MPCMM0002".

Chapter 17

17.0 Simple Network Management Protocol

The RSM supports version 1 (v1) and version 3 (v3) of the Simple Network Management Protocol (SNMP). The RSM can support SNMP queries and send SNMP traps in either v1 or v3 format. The SNMP interface on the RSM very closely mirrors that of the CLI in both syntax and function in that for each MIB object there exists a corresponding CLI dataitem.

Note: Like the CLI, SNMP commands should be executed on the active RSM. The standby RSM responds to commands only if the location parameter is `cmm`.

17.1 Net-SNMP*

The Net-SNMP* open source project is used as the SNMP framework for the RSM. The most important functionalities provided by the Net-SNMP agent are listed below:

- SNMPv3 [*RFC3410*] and SNMPv1 [*RFC1157*] message processing models
- SNMP TRAP v1 [*RFC1215*] and v2 [*RFC3416*]
- UDP transport mapping
- User-based Security Model (USM) [*RFC3414*]
- View-based Access Control Model (VACM) [*RFC3415*]
- Support for atomic execution of SNMP requests

For the full list of Net-SNMP agent features, see: http://www.net-snmp.org.

17.2 Supported MIBs

17.2.1 Chassis Management Module MIB

The RSM comes with RSM MIB (Management Information Base). This is a text file, `MPCMM0003.mib`, that describes the RSM and platform objects to be managed. RSM MIB is not backward compatible with the MIB supported in earlier versions of the RSM firmware. A remote application such as an SNMP/MIB manager can compile and read this file to manage the sensor devices on the RSM, the chassis, and installed blades. Once the RSM firmware has been installed, `MPCMM0003.mib` is located in the `/etc/cmm` directory.

17.2.2 OAM MIB

The RSM comes with a OAM MIB (Management Information Base). This is a text file, `MPCMM0003ext.mib`, that describes new RSM objects related to ShM & OAM API. A remote application such as an SNMP/MIB manager can compile and read this file to manage additional objects on the RSM. Once the RSM firmware has been installed, `MPCMM0003ext.mib` is located in the `/etc/cmm` directory.

17.2.3 MIB II

MIB II module implements MIB II [*RFC1213*] support. This module comes as part of the Net-SNMP* package. The RSM supports the MIB II objects listed in Table 27, “MIB II Objects - System Group” and Table 28, “MIB II - Interface Group”. The writeable objects (those with access read-write) can be set in their respective fields in the `/etc/cmm/netsnmp/snmpd.conf` file. Only the objects described in this section can be customized for the RSM.

Table 27. MIB II Objects - System Group

Object	Syntax	Access	Description
sysDscr	DisplayString	read-only	"Linux *product_name*[a] *kernel_version*[b] *firmware_build_date*[c] armv51"
sysObjectID	OBJECT IDENTIFIER	read-only	iso(1).org(3).dod(6).internet(1).private(4) .enterprises(1).intel(343).products(2).Server-Management(10).Chassis-Management(3).mpcmm0003(2)
sysContact	DisplayString	read-write	String of at most 128 bytes
sysName	DisplayString	read-write	Default string value of "a6k-rsm-j"[d]
sysLocation	DisplayString	read-only	String of at most 128 bytes

a. a6k-rsm-j
b. Version of the Linux kernel
c. Build date of the shelf manager module firmware
d. String matches the product name of the shelf manager module board on which the firmware is running.

Table 28. MIB II - Interface Group

Object	Syntax	Access	Description
ifDscr	DisplayString	read-only	String value of "10/100BASE-TX"

17.3 Use of Sub-FRUs

The MIB includes support for AdvancedMC* (Advanced Mezzanine Cards) and other entities that appear as sub-FRUs of another device. Sub-FRUs are addressed with an appended sub-FRU ID. If a FRU ID is specified, only sensors associated with that FRU ID are returned in response to a query and the FRU ID is prepended to the name of the sensors.

If no sub-FRU ID is specified, all known sensors are displayed in response to a query. The FRU ID associated with each of those sensors is prepended to the name of the sensor in the output.

If no sub-FRU ID is specified when querying location health information, only the highest severity health event for the location and all of its sub-FRUs taken together is returned.

These output format rules are used wherever a sensor name appears, including target listings, SEL dumps, and any alerts.

The Presence and UnHealthyLocations MIB objects are supported for each location. In addition, Presence is also supported for every sub-FRU at a location.

If a CLI command that is valid for *location*:0 is executed using the SNMP interface but with no FRU ID specified, a FRU ID of 0 is assumed. Information only for the FRU with an ID of 0 is read or written at that location.

Note: The FRU numbers used to identify the sub-FRUs is always one greater than the FRU ID. Thus, a blade that has a sub-FRU with a FRU ID of 0 would have a FRU number equal to 1. Similarly, a blade that has a sub-FRU with a FRU ID of 1 would have a FRU number equal to 2, and so on.

17.4 Third-party Chassis Support

The MIB supports the use of the RSM in a various chassis types.

A chassis may house non-intelligent fan trays, PEMs, or air filter trays. An alias for each of these devices must be defined in the `[Alias Output]` section of the `cmm.ini` file. The SNMP daemon running on the RSM requires that the names in these sections be used for the aliases:

- Section 17.4.1, “Fan Tray” on page 84
- Section 17.4.2, “Power Entry Module” on page 84
- Section 17.4.3, “Air Filter Tray” on page 84.
- Section 17.4.4, “Shelf FRU” on page 84
- Section 17.4.5, “SAP” on page 84

17.4.1 Fan Tray

Define the alias(es) `FanTray`*n* where *n* is the instance ID (*not* the FRU ID) of the fronted fan tray. If there are three fan trays, the aliases must be `FanTray1`, `Fantray2`, and `FanTray3`.

Because the numeric suffix following FanTray denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so both the “F” and the “T” in `FanTray`*n* must be capitalized.

17.4.2 Power Entry Module

Define the aliases PEM*n*, where *n* is the instance ID (*not* the FRU ID) of the fronted PEM. If there are two PEMs, the aliases must be PEM1 and PEM2.

Because the numeric suffix *n* in the alias PEM*n* denotes an instance ID, the suffix may not match the FRU ID. Also, these aliases are case-sensitive, so PEM in PEM*n* must be capitalized.

17.4.3 Air Filter Tray

Define the alias `FilterTray`*n* where *n* is the instance ID (*not* the FRU ID) of the fronted air filter tray. These aliases are case-sensitive, so both the “F” and the “T” in `FilterTray`*n* must be capitalized.

Note: There can be only one fronted filter tray in the chassis.

17.4.4 Shelf FRU

Define the aliases `ShelfFru`*n*, where *n* is the instance ID (not the FRU ID) of the fronted Shelf Fru. If there are 2 Shelf Fru's, the aliases must be `ShelfFru1` and `ShelfFru2`.

Because the numeric suffix following `ShelfFru` denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so both the "S" and the "F" in `ShelfFru`*n* must be capitalized.

17.4.5 SAP

Define the aliases SAP*n*, where *n* is the instance ID (not the FRU ID) of the fronted Shelf Alarm Panel. If there are 2 SAP's, the aliases must be SAP1 and SAP2.

Because the numeric suffix following SAP denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so all three letters "S","A"and the "P" in SAP*n* must be capitalized.

Note: If there is only one fronted SAP then *n* should be omitted and the alias should be SAP.

17.4.6 Alias Mappings

The alias entries in the section [Alias Output] of the cmm.ini file provide linkage between alias names and FRU IDs.

17.5 SNMP Agent

The SNMP agent (snmpd) listens to SNMP v1 queries (gets and sets) by default, evokes the corresponding MIB Module to process the request, and sends the SNMP response with return data to the SNMP/MIB manager. The agent can also be configured to respond to v3 queries. The SNMP agent in the RSM is implemented to support SNMP get, SNMP get next, and SNMP set for all supported MIB objects.

All SNMP set queries are logged in the command log file, user.log.

17.5.1 Configuration Files

The SNMP Agent configuration is stored in /etc/cmm/netsnmp/snmpd.conf configuration file. This configuration file is managed directly by the user.

For more information regarding SNMP configuration and the snmpd.conf file, read the manual page for the file at:

http://www.net-snmp.org/man/snmpd.conf.html

The SNMP agent can be configured to support SNMPv1 or SNMPv3. There are two initial configuration files available:

/etc/cmm/netsnmp/snmpdv1.conf - a sample configuration file for the SNMP agent running SNMPv1. To activate this configuration, copy this file to /etc/cmm/netsnmp/snmpd.conf.

/etc/cmm/netsnmp/snmpdv3.conf - a sample configuration file for the SNMP agent running SNMPv3. To activate this configuration, copy this file to /etc/cmm/netsnmp/snmpd.conf.

17.5.2 Configuring SNMP Agent Port

The SNMP agent is set up to use port 161 by default. The agent can be configured to use a different port by adding the following line to the /etc/cmm/netsnmp/snmpd.conf file:

agentaddress *port_number*

17.5.3 Configuring Agent to Respond to SNMP v3 Requests

Initially, the SNMP agent is configured to run SNMP v1 but it can be reconfigured at any time to run SNMP v3. SNMP v3 adds support for strong authentication and private communication.

To change the SNMP agent to respond to SNMP v3 queries:

1. Copy /etc/cmm/netsnmp/snmpdv3.conf to /etc/cmm/netsnmp/snmpd.conf by executing this command:

   ```
   cp /etc/cmm/netsnmp/snmpdv3.conf /etc/cmm/netsnmp/snmpd.conf
   ```

2. Restart the snmpd agent by executing the following command:

   ```
   kill -s SIGHUP ‘pidof snmpd‘
   ```

17.5.4 Configuring Agent Back to SNMP v1

To reconfigure the agent back to SNMP v1, follow the same steps as above substituting `/etc/cmm/netsnmp/snmpdv1.conf` for `/etc/cmm/netsnmp/snmpdv3.conf`. as follows:

```
cp /etc/cmm/netsnmp/snmpdv1.conf /etc/cmm/netsnmp/snmpd.conf
```

17.5.5 Setting up SNMP v1 MIB Browser

By default, the community name for the SNMP agent on the RSM is `public` for both read and write. This can be changed by editing the `/etc/cmm/netsnmp/snmpd.conf` file on the RSM and then signalling the SNMP daemon to re-read the file by executing this command:

```
kill -SIGHUP ‘pidof snmpd‘
```

Note: The SNMP MIB browser needs to match the community name for both reads and writes.

17.5.6 Setting up an SNMP v3 MIB Browser

To manage the RSM using an SNMP v3 MIB browser or manager, configure the browser with the following parameters:

1. Load and compile the `MPCMM0003.mib` and `MPCMM0003ext.mib` files
2. Set the SNMP v3 security parameters:
   - Set SNMP v3 agent user
     At default, User: `root`
   - Set the MD5 Authentication password: `cmmrootpass`
   - Set the DES Encryption password: `cmmrootpass`

17.5.7 Changing the SNMP MD5 and DES Passwords

To change the MD5 Authentication and DES Encryption passwords for the SNMP interface on the RSM, use one of the following methods:

Method 1

1. Edit `/etc/cmm/netsnmp/snmpd.conf` on the active RSM and add the following line:

   ```
   createUser root MD5 cmmrootpass DES cmmrootpass
   ```

   This line allow the creation of user root with MD5 authentication password as `cmmrootpass`, and DES encryption password as `cmmrootpass`.
2. Add more lines for more users if needed.
3. Restart the SNMP agent.

Method 2

Use the `snmpusm` utility from a Linux* host that has net-snmp packet install. You can learn more at http://www.net-snmp.org.

17.6 SNMP Traps

The RSM sends SNMP trap messages to a remote application regarding any abnormal system events. When enabled, the RSM will issue SNMP v1 traps on port 162. The RSM can also be configured to issue SNMP v3 traps. Other SNMP trap parameters, such as version, port, community, format, or addresses can also be configured.

SNMP trap parameters can be set only on the active RSM. Attempting to set these parameters on the standby RSM will result in an error.

17.6.1 SNMP Trap Format

All SNMP traps generated by the RSM adhere to one of the following formats:

- proprietary format
- "Platform Event Trap Format Specification"

SNMP traps can be sent in a proprietary format or in PET format.

17.6.2 Proprietary SNMP Trap Format

The first four items (Time, Location, Chassis Serial #, and Board) constitute the header and are always sent. This information that does not necessarily come from the event itself. These pieces of information are helpful in tracing the trap back to its source.

17.6.2.1 Proprietary SNMP Trap Header Format

`Time :` *TimeStamp* `, Location :` *ChassisLocation* `, Chassis Serial # :` *ChassisSerialNumber* `, Board :` *Location*

- *TimeStamp* is in the format [Day] [Month] [Date] [HH:MM:SS] [Year]. For example, the timestamp might be Thu Apr 14 22:20:03 2005
- *ChassisLocation* is the chassis location information recorded in the chassis FRU
- *ChassisSerialNumber* is the chassis serial number recorded in the chassis FRU
- *Location* indicates where the sensor generating the event is located (for example, RSM)

The next portion can be controlled by a RSM variable to turn it on or off. This section provides the text interpretation of the event.

17.6.2.2 Proprietary SNMP Trap Text Translation Format

`Sensor :` *SDRSensorName* `, Event :` *HealthEventString* `, Event Code :` *EventCodeNumber*

- *SDRSensorName*: The name given to the sensor in the Sensor Data Record (SDR).
- *HealthEventString*: The RSM's translation of the event.
- *EventCodeNumber*: A hexadecimal number that uniquely defines the event. The format of the event code is 0x*NNNN*, where N is a hexadecimal digit.

17.6.2.3 Proprietary SNMP Trap Raw Data Format

The final portion that an SNMP trap message might include is the "raw" portion of the trap. This data reports the original sixteen bytes of the system event as ASCII upper case hex bytes.

```
Raw Hex : [ 12 34 56 78 9A 0C 33 81 F2 1B 39 42 DE 64 BA 88 ]
```

Note: The sixteen bytes of raw hex data shown are an example. The actual data will be different.

17.6.3 Configuring SNMP Trap Format

To configure the SNMP trap format, execute this command:

```
cmmset -d SNMPTrapFormat -v <format>
```

where <format> is one of

- legacy Text
- legacy Raw
- legacy Text&Raw
- PET

To configure the SNMP trap format per trap address, execute this command:

```
cmmset -d SNMPTrapFormat<index> -v <format>
```

<index> is the number of the trap address (1–5) being set, <format> is defined as above.

The following figures show what the output looks like depending on the setting of the `snmptrapformat` dataitem.

snmptrapformat = 1

`Time : `*TimeStamp*` , Location : `*ChassisLocation*` , Chassis Serial # : `*ChassisSerialNumber*` , Board : `*Location*` , Sensor : `*SDRSensorName*` , Event : `*HealthEventString*` , Event Code : `*EventCodeNumber*

snmptrapformat = 2

`Time : `*TimeStamp*` , Location : `*ChassisLocation*` , Chassis Serial # : `*ChassisSerialNumber*` , Board : `*Location*` , Raw Hex : `*16_bytes_of_hex_data*

snmptrapformat = 3

`Time : `*TimeStamp*` , Location : `*ChassisLocation*` , Chassis Serial # : `*ChassisSerialNumber*` , Board : `*Location*` , Sensor : `*SDRSensorName*` , Event : `*HealthEventString*` , Event Code : `*EventCodeNumber*` , Raw Hex : `*16_bytes_of_hex_data*

snmptrapformat = 4

PET format [“Platform Event Trap Format Specification”]

17.6.4 Configuring the SNMP Trap Port

To configure the SNMP trap port to a different port number, execute the following command:

```
cmmset -l cmm -d SNMPTrapPort -v <port_number>
```

port_number is the desired SNMP trap port number.

17.6.5 Configuring RSM to Send SNMP v3 Traps

If the SNMP trap version has not been set using the `SNMPTrapVersion` dataitem in the CLI the firmware will default to Trap Version 3.

To configure the RSM to send SNMP v3 traps, execute this command:

```
cmmset -l cmm -d SNMPTrapVersion -v v3
```

17.6.6 Configuring RSM to Send SNMP v1 Traps

To configure the RSM to send SNMP v1 traps, execute this command:

```
cmmset -l cmm -d SNMPTrapVersion -v v1
```

17.7 Configuring and Enabling SNMP Trap Addresses

The RSM allows up to five SNMP trap addresses, namely, SNMPTrapAddress1-5.

When the RSM is configured to send SNMP v3 traps, it is recommended that only one SNMPTrapAddress be configured because of the large number of traps that can be generated on a loaded system.

Note: In redundant RSM systems, SNMP Trap Address 1 *must* be set to a valid IP address on the network that the RSM can ping. This is used as a test of network connectivity as well as being the first SNMP Trap Address.

17.7.1 Configuring SNMP Trap Addresses

To configure an SNMP trap address, execute this command:

```
cmmset -l cmm -d SNMPTrapAddress<index> -v ip_address
```

<index> is the number of the trap address (1–5) that is being set, and *ip_address* is the IP address of the trap receiver.

17.7.2 Enabling and Disabling SNMP Traps

SNMP trap addresses are disabled by default.

To enable SNMP traps, execute the following command:

```
cmmset -l cmm -d SNMPEnable -v enable
```

To disable SNMP traps, execute the following command:

```
cmmset -l cmm -d SNMPEnable -v disable
```

To check the status of SNMP traps, execute the following command:

```
cmmget -l cmm -d SNMPEnable
```

17.7.3 Alerts Using SNMP v3

To receive the SNMP v3 trap, the remote application, such as the trap listener, needs to:

1. Set the SNMP v3 trap user. The default trap user is `root`.
2. Set the MD5 Authentication password. The default MD5 Authentication password is `publiccmm`.
3. Set the DES Encryption password. The default DES Encryption password is `publiccmm`.

Note: To change the passwords (MD5 and DES) for the SNMP v3 trap, change the SNMP Trap Community string from the CLI interface by executing the following command on the active RSM:

```
cmmset -d snmpTrapCommunity -v <community>
```

You can also change the SNMP Trap Community string from the SNMP manager console.

17.8 Configuring SNMP Trap Acknowledgement

SNMP trap acknowledgement status controls RSM behavior with respect to transmitted SNMP traps in PET format.

To configure SNMP trap acknowledgements, execute this command:

```
cmmset -d SNMPTrapAcknowledge<index> -v <status>
```

where <status> is one of:

- enabled - Alert is assumed successful only if acknowledged is returned.
- disabled - Alert is assumed successful if transmission occurs without error.

Note: Legacy trap format does not support acknowledgements.

17.9 Configuring SNMP Trap Retries

The process of sending SNMP traps is configurable.

To configure the number of SNMP trap send retries, execute this command:

```
cmmset -d SNMPTrapRetryCount<index> -v <count>
```

To configure the time between automatic retries, execute this command:

```
cmmset -d SNMPTrapRetryInterval<index> -v <interval>
```

17.10 Sending SNMP Traps for Unrecognized Events

If dataitem SNMPSendUnrecognizedEvents is set to 1, the RSM sends SNMP traps for unrecognized events. The default value of this dataitem is 0.

To configure the RSM to send SNMP traps for unrecognized events, execute this command:

```
cmmset -d SNMPSendUnrecognizedEvents -v <state>
```

Table 29. Results of Dataitem Settings

	SNMPTrapFormat Control		
Recognized Event	1 (text)	2 (raw)	3 (text&raw)
	Header and text	Header and raw data	Header, text, and raw data. Helps in cases where the event is partially translated in the text portion.
Unrecognized Event	SNMPSendUnrecognizedEvents = 0 No trap message sent		
	SNMPSendUnrecognizedEvents = 1 Useful in allowing you to see that there are unrecognized events. However, it does not give enough information to understand the event.	Header and raw data	Header, text, and raw data. The Text portion simply states that the RSM could not translate the event.

17.11 Trap Connect Sensor

The “Trap Connect” sensor tracks trap connectivity. For a detailed description, see Appendix D, “OEM Sensor Events”.

17.12 SNMP Security

This section describes SNMP security features for SNMP v1 and SNMP v2.

17.12.1 SNMP v1 Security

SNMP v1 utilizes the community name for authentication. If the SNMP manager/client sends a request message containing a community name that does not match the community name set in the SNMP agent, the agent responds with an authentication failure message.

Caution: The community name is not encrypted during transmission.

17.12.2 SNMP v3 Security Authentication and Privacy Protocol

The RSM supports the highest security level for SNMP v3. MD5 is used for the authentication protocol and DES is used for the privacy protocol. When in this mode, you need to specify each password (authKey, privKey) for these protocols. The SNMP v3 packet is securely encrypted during transmission. This is the default security level of the RSM when configured for SNMP v3.

The fields listed in Table 30, “SNMP v3 Security Fields for Traps” and Table 31, “SNMP v3 Security Fields for Queries”are defined to handle all SNMP v3 security levels.

Table 30. SNMP v3 Security Fields for Traps

Security Name	User Name	Default Value:
SecurityName	User name	root
AuthProtocol	authentication type	MD5
AuthKey	authentication password	publiccmm
PrivProtocol	privacy type	DES
PrivKey	privacy password	publiccmm

Table 31. SNMP v3 Security Fields for Queries

SecurityName	User Name	Default Value:
SecurityName	User name	root
AuthProtocol	authentication type (MD5)	MD5
AuthKey	authentication password	cmmrootpass
PrivProtocol	privacy type (DES)	DES
PrivKey	privacy password	cmmrootpass

17.13 Additional Notes

This section contains additional information about SNMP and the MIB.

17.13.1 Redundant ListDataItems MIB Objects

The SNMP MIB contains some objects named "*xxx*ListDataItems" (for example, cmmFruListDataItems). These objects return the dataitems available *using the CLI* (not SNMP) for a particular target or location. The target or location is indicated by the portion of the MIB tree in which the MIB object is located.

Not every possible target or location available in the CLI has a corresponding "*xxx*ListDataItems" object in the SNMP MIB. These objects provide information beyond the scope of SNMP and are not needed to perform SNMP operations.

Chapter 18

18.0 Remote Management Control Protocol

The Remote Management Control Protocol (RMCP) has been defined by the Distributed Management Task Force (DMTF) for supporting pre-OS and OS-absent management. RMCP uses a simple request-response protocol that can deliver IPMI messages using UDP datagrams. RMCP is defined in "Alert Standard Format (ASF) Specification version 2.0".

The RMCP+ stack implements the Remote Management Control Protocol Plus (RMCP+) as described in "Intelligent Platform Management Interface Specification v2.0".

In addition to full support for IPMI 2.0, this implementation of RMCP+ is backward compatible with RMCP (as described in "Intelligent Platform Management Interface Specification v1.5") and provides the following services (as described in "Intelligent Platform Management Interface Specification v2.0"):

- RMCP+ message processing
- ASF presence ping/pong messages processing
- RMCP+ integrity, authentication, and encryption algorithms:
- Authentication algorithms supported: RAKP-none, RAKP-HMAC-SHA1, and RAKP-HMAC-MD5
- Integration algorithms supported: None, HMAC-SHA1-96, HMAC-MD5-128, and HMAC-SHA1-128
- Encryption algorithms supported: None and AES-CBC-128

In addition, RMCP+ can be configured to use SCTP instead of UDP as a transport protocol to provide a reliable transport option. Note, however, that this is a custom extension that is not compatible with RMCP+ as defined in "Intelligent Platform Management Interface Specification v2.0".

18.1 RMCP Client and Server Communication

RMCP messages are sent using UDP datagrams over the Ethernet. The RMCP server communicates on management port 623 for handling RMCP requests. This is the primary RMCP port. A secondary port, 664, is used when encryption is necessary for security.

Note: The implementation of the RMCP server provided with the RSM firmware package listens for RMCP packets only on port 623 (the primary RMCP port).

When an RMCP packet arrives, the RMCP server checks the packet. If it is an invalid version or not a valid IPMI RMCP packet, the server drops the packet. If the session data in the packet is invalid, not available, duplicated, or out of order, or slots are full, the server returns an RMCP error message to the RMCP client. Otherwise, the server decodes the RMCP message.

If the message is the RMCP "ping" message, the server returns the RMCP "pong" message to indicate to the client that it has successfully found an RMCP server. If the RMCP packet contains a valid message other than "ping", the message is forwarded through the RSM interface to the destination indicated in the message. If the RSM receives an appropriate IPMI response from the final destination, the RSM returns the IPMI response in a properly formatted RMCP message back to the RMCP server, which then returns the message to the RMCP client over the network.

18.2 RMCP Modes

The RMCP server on the RSM may be configured to operate in one of two modes shown in Table 32, "RMCP Modes". The configuration flag is located in `shm.conf` configuration file and is read on system startup.

Table 32. RMCP Modes

RMCP Mode	Description
Enabled	The RMCP feature functionality is fully operational and a RMCP client can initiate a session regardless of the host /server power state and operating system health. This is the default system setting.
Disabled	Disables the RMCP functionality. In this mode the RMCP server discards the requests it receives over the network.

18.3 Enabling and Disabling RMCP

To determine whether RMCP is enabled or disabled, execute the following command:

```
cmmget -l cmm -d RMCPEnable
```

The CLI returns 1 if RCMP is enabled or 0 if RMCP is disabled.

To enable or disable RMCP, execute the following command:

```
cmmset -l cmm -d RMCPEnable -v <switch>
```

switch is either 0 to disable or 1 to enable.

Note: If RMCP is already enabled, executing the command to enable RMCP returns the message `IMB ERROR Completion Code`. In this situation the message can be safely ignored.

18.4 RMCP Discovery

According to the IPMI Specification Version 1.5, the RMCP client uses Ping/Pong messages to discover the existence of an RMCP server. The RMCP server supports the discovery mechanism with two messages:

- RMCP/ASF Presence Ping message
- RMCP/ASF Pong message

In the Pong message, the RSM communicates the following information:

- IANA Enterprise number
- Supported Entities: IPMI supported and Alert Standard Format version 1.0

18.5 IPMB Slave Addresses

The embedded IPMI message within a RMCP message needs to have IPMB slave address set. The slave address required by this protocol should be set to 20h to address the BMC. On the other hand, the RMCP client may use any of the addresses shown in Table 33, “RMCP Slave Addresses” as its slave address. However, only *even* values are allowed, that is, the least significant bit of the slave address must always be zero.

Table 33. RMCP Slave Addresses

Nodes	Value
RMCP Server Slave Address	20h
RSM1 RMCP Server Slave Address	10h[a]
RSM2 RMCP Server Slave Address	12h[a]
RMCP Client Slave Address	C0h-CEh

a. Actual address is derived from the hardware address for the RSM in the chassis where the RSM is installed. The values in this table are provided only as examples.

18.6 Communicating with RMCP Server on RSM

To communicate with the RSM's RMCP server, an RMCP client must do the following:

- Provide the RMCP server's IP address
- Provide a user name, which is initially set to `root`
- Provide a user password, which is initially set to `cmmrootpass`
- Turn RMCP on

18.7 RMCP Security

18.7.1 RMCP User Privilege Levels

The following privilege levels defined in "Intelligent Platform Management Interface Specification v1.5" are supported (ordered from most restrictive to least restrictive privilege):

1. User level (most restrictive)
2. Operator level
3. Administrator level (least restrictive)
4. OEM Proprietary level (configurable)

The RMCP server provides the user and password support associated with these privilege levels. Each command requires a certain privilege level. Commands that require a higher privilege level than the one associated with the user issuing the command cannot be executed.

The user name, password, and privilege level can be set using CLI commands defined in Section 13.2, "User Management" on page 76.

Note: Only the user name `root` is supported by the RSM firmware.

18.7.2 RMCP Maximum Privilege Levels

The following CLI command is used to set the maximum allowed privilege level for channel access:

```
cmmset -t Channel:<channel#> -d MaxPrivLevel -v <level>
```

Currently it is possible to configure privilege level only for the IPMI LAN channel. The following CLI command is used to get the maximum allowed privilege level for channel access:

```
cmmget -t Channel:<channel#> -d MaxPrivLevel
```

18.7.3 Configuring IPMI Command Privileges

Each time some IPMI command is called, RMCP checks if the caller has sufficient privileges to use this command. To do so, RMCP consults the IPMI privileges table.

Privilege levels for administrator, operator, and user and fixed and not subject to changes. In contrast, for the OEM privilege level, the user may decide which IPMI messages can be executed on this level. The RSM provides a CLI interface to set the OEM privilege level for an IPMI function.

To set the OEM privilege level for an IPMI function, execute the command:

```
cmmset -l cmm -t RmcpFunc<netfn>:<cmd> -d OemPermission -v
{0|disable|1|enable}
```

The `rmcp.conf` file located in the `/etc/cmm` directory of the RSM stores the configuration of OEM privileges allowed for each IPMI command on the RSM. The format of a single entry is as follow:

```
NetFunNUMCmdNUM = 'enable'
```

NetFunNUMCmdNUM keyword identifies the specific IPMI command. The NUM in the keyword should be replaced by the appropriate IPMI command NetFun or Cmd numeric code.

The RSM does not use the `cmdPrivillege.ini` file.

18.7.4 BMC Key

IPMI v1.5 uses a single key (the user key/password) that is used both for authentication and in integrity (AuthCode) calculations. IPMI v2.0/RMCP+ can be configured to use a single key ("onekey") login where the user key is used both for authentication and to generate a Session Integrity Key that is used in integrity (AuthCode) calculations, or a "two-key" login where the user key is used for authentication, and a separate "BMC key", KG, is used to create the Session Integrity Key that is used in integrity (AuthCode) calculations.

The following CLI command is used to set BMC key:

```
cmmset -t Channel:<channel#> -d BmcKey -v <key>
```

The following CLI command is used to get BMC key:

```
cmmget -t Channel:<channel#> -d BmcKey
```

18.7.5 Authentication

The following CLI command is used to set authentication types:

```
cmmset -t Level:<level> -d AuthTypes -v <type>[,<type>]
```

where `<level>` is one supported user privilege levels listed in Chapter 18.0, "RMCP User Privilege Levels" on page 95 and `<type>` is one of none, straight, md2, md5.

The following CLI command is used to get authentication types:

```
cmmget -t Level:<level> -d AuthTypes
```

18.7.6 IPMI System GUID

As per the IPMI specification, the RSM is assigned a globally unique ID (GUID) for the system to support the remote discovery process and other operations (e.g. SNMP traps in PET format). This RSM configuration parameter is stored in the `/etc/cmm/rmcp.conf` file.

18.8 RMCP over SCTP Transport

"Intelligent Platform Management Interface Specification v2.0" defines UDP as the transport protocol for RMCP packets. SCTP has been added as an optional transport protocol for RMCP. SCTP is a modern transport protocol standardized in IETF. It was designed to meet the requirements of the growing IP telecommunication market to facilitate transporting various telecommunication signaling protocols over the Internet. SCTP is connection-oriented and offers greater reliability than older protocols like UDP or TCP. SCTP and UDP use the same port number (623) for RMCP+.

To select a transport option for RMCP, execute the command:

```
cmmset -l cmm -d RmcpTransport -v {udp|sctp}
```

To get the currently used transport protocol used by RMCP, execute the command:

```
cmmget -l cmm -d RmcpTransport
```

18.9 Supported IPMI Commands

The IPMI commands listed in Table 34, "IPMI Commands Supported by RSM RMCP" are the ones supported by the RSM when sent to it using RMCP. To configure privileges for the commands see Section 18.7.3, "Configuring IPMI Command Privileges" on page 95.

Note: If an IPMI command does not appear in Table 34, it cannot be executed using RMCP and will be rejected.

Table 34. IPMI Commands Supported by RSM RMCP (Sheet 1 of 3)

Command Type	Where Defined	Command	Available on IPMB Address
IPMI Device Global	"Intelligent Platform Management Interface Specification v1.5"	Get Device ID	(Active ShM address, LUN 00), (RSM HW address, LUN 00)
		Get Self Test Results	
BMC Device and Messaging Commands	"Intelligent Platform Management Interface Specification v1.5"	Send Message	(Active ShM address, LUN 00)
		Get Channel Authentication Capabilities	
		Get Session Challenge	
		Activate Session	
		Set Session Privilege Level	
		Close Session	
		Get Session Info	
		Get AuthCode	
		Set Channel Access	
		Get Channel Access	
		Get Channel Info	
		Set User Access	
		Get User Access	
		Set User Name	
		Get User Name	
		Set User Password	
Chassis Device Commands	"Intelligent Platform Management Interface Specification v1.5"	Get Chassis Capabilities	(Active ShM address, LUN 00)
		Get Chassis Status	
		Chassis Control	
Event Commands	"Intelligent Platform Management Interface Specification v1.5"	Get Event Receiver	(Active ShM address, LUN 00), (RSM HW address LUN 00), (RSM HW address LUN 02)
		Set Event Receiver	
		Platform Event	(Active ShM address, LUN 00)
PEF and Alerting Commands	"Intelligent Platform Management Interface Specification v1.5"	Get PEF Capabilities	(Active ShM address, LUN 00)
		Set PEF Configuration Parameters	
		Get PEF Configuration Parameters	
		PET Acknowledge	

Table 34. IPMI Commands Supported by RSM RMCP (Sheet 2 of 3)

Command Type	Where Defined	Command	Available on IPMB Address
Sensor Device Commands	"Intelligent Platform Management Interface Specification v1.5"	Get Device SDR Info	(Active ShM address, LUN 00), (RSM HW address LUN 00), (RSM HW address LUN 02)
		Get Device SDR	
		Reserve Device SDR Repository	
		Get Sensor Hysteresis	
		Get Sensor Threshold	
		Get Sensor Event Enable	
		Re-arm Sensor Events	
		Get Sensor Event Status	
		Get Sensor Reading	
FRU Device Commands	"Intelligent Platform Management Interface Specification v1.5"	Get FRU Inventory Area Info	(Active ShM address, LUN 00), (RSM HW address LUN 00)
		Read FRU Data	
		Write FRU Data	
SDR Repository Commands	"Intelligent Platform Management Interface Specification v1.5"	Get SDR Repository Info	(Active ShM address, LUN 00)
		Reserve SDR Repository	
		Get SDR	
		Partial Add SDR	
		Delete SDR	
		Clear SDR Repository	
		Get SDR Repository Time	
SEL Device Commands	"Intelligent Platform Management Interface Specification v1.5"	Get SEL Info	(Active ShM address, LUN 00)
		Reserve SEL	
		Get SEL Entry	
		Add SEL Entry	
		Clear SEL	
		Get SEL Time	
		Set SEL Time	
LAN Device Commands	"Intelligent Platform Management Interface Specification v1.5"	Set LAN Configuration Parameters	(Active ShM address, LUN 00)
		Get LAN Configuration Parameters	

Table 34. IPMI Commands Supported by RSM RMCP (Sheet 3 of 3)

Command Type	Where Defined	Command	Available on IPMB Address
AdvancedTCA*	"PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification"	Get PICMG Properties	(Active ShM address, LUN 00), (RSM HW address LUN 00)
		Get Address Info	
		Get Shelf Address Info	(Active ShM address, LUN 00)
		Set Shelf Address Info	
		FRU Control	(Active ShM address, LUN 00), (RSM HW address LUN 00)
		Get FRU LED Properties	
		Get LED Color Capabilities	
		Set FRU LED State	
		Get FRU LED State	
		Set IPMB State	
		Set FRU Activation Policy	
		Get FRU Activation Policy	
		Set FRU Activation	
		Get Device Locator Record ID	
		Get Port State	
		Compute Power Properties	
		Set Power Level	
		Get Power Level	
		Renegotiate Power	(Active ShM address, LUN 00)
		Get Fan Speed Properties[a]	
		Set Fan Level[b]	
		Get Fan Level[c]	
		Get IPMB Link Info	(Active ShM address, LUN 00), (RSM HW address LUN 00)
	"Intelligent Platform Management Interface Specification v2.0"	Open Session Request	(Active ShM address, LUN 00)
		Open Session Response	
		RAKP 1	
		RAKP 2	
		RAKP 3	
		RAKP 4	
		Set Channel Security Keys	
		Get Channel Cipher Suits	

a. Applies only to fan trays fronted by the Chassis Management Module.
b. Applies only to fan trays fronted by the Chassis Management Module.
c. Applies only to fan trays fronted by the Chassis Management Module.

18.10 Completion Codes for RMCP Messages

Table 35, "RMCP Message Completion Codes" lists the completion codes for RMCP messages. See "Intelligent Platform Management Interface Specification v1.5" for more information.

Table 35. RMCP Message Completion Codes

Code	Description
00	Success
C0	Busy
C1	Invalid Command
C2	Command invalid for a given LUN
C7	Request data length invalid
C8	Requested data field length limit exceeded. (too long)
C9	Requested Offset (in the data) Out of Range
CB	Not Found
CC	Invalid field in the Request
CD	Illegal Command
10	RMCP Session/User Authentication Failed
11	RMCP Session Active
12	RMCP Session in Authentication Phase

Chapter 19

19.0 IPMI Pass-Through

19.1 Overview

The Intelligent Platform Management Interface (IPMI) pass-through feature allows IPMI commands to be sent directly to any device in the chassis through the RSM without being processed by lower layers of the RSM software. The command can be sent over the CLI, SNMP, or ShM API. The command is sent even if the blade or device appears to the RSM to not be present or not able to communicate using IPMI.

Note: A blade can appear to not be present even if it is physically in the chassis because the state of the blade is determined through communication between the blade and the RSM. For example, if you insert a blade but do not close the latch, the blade will not be marked as present since no message was sent to the RSM to notify it of the state transition of the blade from M1 to M2.

19.2 Command Syntax

This syntax of this command is:

```
cmmset -l <location> -d IPMICommand -v <command_request_string>
```

Specify the *location* to which the IPMI command is to be sent. The possible values of *command_request_string* are described in the following sections.

19.2.1 Command Request String Format

This command request string contains the data for the command to be sent. It has the following format:

netfn [*lun*] *cmd* [*data_0 …. data_n*]

netfn: A decimal or hexadecimal number specifying the Net Function of the IPMI request. The number must be an even integer greater than or equal to 0 and less than 62.

lun: A decimal or hexadecimal number specifying the destination LUN (logical unit) of the IPMI request. This number must be an integer greater then or equal to 0 and less than or equal to 3. The number must also be immediately preceded by the uppercase or lower case letter L (for example, L3 or l3). This argument is optional and defaults to L0 if not provided.

cmd: A decimal or hexadecimal number specifying the command number of the IPMI request. The number must be an integer greater than or equal to 0 and less than or equal to 255.

data_0 …. data_n: Decimal or hexadecimal numbers separated by spaces specifying the IPMI request data. These numbers must be integers greater than or equal to 0 and less than or equal to 255. There can be at most 25 data items in this list.

Hexadecimal numbers are written beginning with 0x followed by the hexadecimal digits of the number.

The request string is checked for the format and ranges specified above. Any further checking of the command or data is left up to the receiver. If the range or format checking fails, the error code E_CLI_INVALID_SET_DATA is returned.

Note: See "Intelligent Platform Management Interface Specification v1.5" for further details on IPMI commands and the values described above.

19.3 Response String

If transmission of the command is successful, a string of data is returned as the response to the IPMI request. All data values are decimal integers separated by spaces. At least one number is always returned, namely, the completion code of the command. The number and meaning of the other numbers in the response string depend on the command sent.

If the transmission of the command fails, the error E_WP_I2C_ERROR is returned by the CLI.

Note: Not all commands return a response after being successfully transmitted. If the CLI receives no response before the timeout expires, the CLI returns an error.

19.4 Usage Examples

This section presents examples of sending IPMI commands using the CLI, SNMP, and ShM API.

19.4.1 Using the CLI

Send an AdvancedTCA Get PICMG Properties command to LUN 0 of the RSM:

```
# cmmset -l cmm -d IPMICommand -v "0x2c L0 0 0"

0 0 18 0 0
```

19.4.2 Using ShM API

ShM API function shmMessageSend can be used to send IPMI commands directly to any device in the chassis through the RSM.

19.4.3 Using SNMP

Because the SNMP set command cannot return data, the IPMI pass-through functionality is split into two SNMP objects under each location: IPMICommandReq and IPMICommandRes.

IPMICommandReq is a Read-Write object. After executing a read (get), it returns a string (initially empty) that contains the last successful request performed using SNMP. After executing a write (set) it returns whether the IPMI command was successfully sent and the response was successfully received.

IPMICommandRes will be Read-Only and will return the response string of the last successful IPMICommand. In order to differentiate between requests, the response string will also be followed by the request string separated by “#”.

Send IPMI Get Device ID request to the RSM:

```
# snmpget […] […].cmmIPMICommandRequest
[…].cmmIPMICommandRequest=""

# snmpget […] […].cmmIPMICommandResponse
[…].cmmIPMICommandResponse=""

# snmpset […] […].cmmIPMICommandRequest s "6 1"
OK

# snmpget […] […].cmmIPMICommandRequest
[…].cmmIPMICommandRequest="6 1"

# snmpget […] […].cmmIPMICommandResponse
[…].cmmIPMICommandResponse="0 32 129 5 2 81 255 87 1 0 65 8 0 0 0 0 # 6 1"
```

Chapter 20

20.0 RSM Scripting

20.1 Command Line Interface Scripting

In addition to calling the Command Line Interface (CLI) directly, commands can be called through scripts using bash shell scripting. These scripts can be used to create a single command from several CLI commands or to give more detailed information.

For example, you may want to display all of the fans and their speeds in the chassis. A script could be written that would first call the CLI to find out what fan trays are present. Next, it would find out what fan sensors are in each fan tray. Finally, it would call the CLI to get the current speeds of each of the fans.

Scripts can be written directly using a text editor (`vi`) on the RSM and should be saved on the RSM as a file in flash memory in the `/usr/share/cmm/scripts` directory. Each script must have bash marker `#!/bin/sh` in the first line and have execute permission set for the owner.

20.2 Event Scripting

Health events triggered on the RSM can be used to execute scripts stored locally. Any level of an event can be used as a trigger: normal, minor, major, and critical. Specific event codes can also be used to trigger scripts.

There is a many-to-many relationship between events and scripts. One script can be associated with many events. Conversely, a particular event can be associated with more than one script (e.g., a default script and a user defined script). On the other hand, when the event occurs, RSM launches one and only one script that fits best to event description.

20.2.1 Triggering Scripts from Health Events

The CLI command for associating a script with a health event is (all on one line):

```
cmmset -l <location> -t <target> -d <action type> -v [<time>:]<script>
[args]
```

location is the component in the chassis that the health event is associated with.

target is the sensor to be triggered on.

action_type is `NormalAction`, `MinorAction`, `MajorAction`, or `CriticalAction` depending on the severity of the event to be triggered on.

time (optional) is the script maximum execution time in seconds. The default value is unlimited time.

script is the script file to be run, including parameters to be sent to the script. The script and parameters should be enclosed in quotes. The *script* argument can be the name of the file that contains the script, a relative pathname (one that begins with a directory name and does not begin with "/"), or an absolute pathname beginning with "/".

args (optional) stands for arguments passed to the script.

If you specify the absolute pathname, the `cmmset` command looks for the specified file. If you specify a relative pathname, the `cmmset` command prepends the path `/usr/share/cmm/scripts` directory to create the absolute pathname and then looks for the file using this pathname. If you specify just the filename, `cmmset` assumes the script is located in the `/usr/share/cmm/scripts` directory and looks for it there.

This setting gets written to the `/etc/cmm/policy.conf` file and is synchronized to the standby RSM. It is persistent across boots.

For example, if you want to run a blade powerdown script called “bladepowerdown” stored in the /usr/share/cmm/scripts directory and runs when the ambient temperature triggers a major event for blade 4, the command is:

```
cmmset –l blade4 –t "0:Ambient Temp" –d MajorAction –v "bladeovertemp 4"
```

Note: This assumes that blade4 has a sensor named Ambient Temp on the blade, itself. Consult the appropriate documentation for the blade or other device to learn about the sensors available for that device.

In this example, the /usr/share/cmm/scripts/bladeovertemp script is executed with “4” as the single argument when the Ambient Temp sensor on blade 4 generates a major health event.

You can verify the pathname of the script associated with a particular event and sensor by entering the following command:

```
cmmget –l blade4 -t "0:Ambient Temp" –d MajorAction
```

The output of this command is the absolute pathname of the script (if any) associated with the specified event and sensor, namely in this case:

```
/usr/share/cmm/scripts/bladeovertemp.sh
```

An additional tag (WILDCARD) is added on output to the script name when a particular script association holds for more than one location.

If you attempt to associate a script that does not exist or for which you specify an incorrect pathname, the following error message is returned.

Action Scripts: File *pathame_of_file* Not Found Error. No Association has been made.

Error checking on the cmmset command applies both to the values supplied with the command and to values stored in the /etc/cmm/policy.conf file.

20.2.2 Triggering Scripts from Event Codes

The RSM allows scripts to be associated with specific events that may not necessarily be health related, such as the assertion of a threshold sensor. This allows any single event that can occur on the RSM to have an associated script. To allow the user to set scripts based on any event, a unique event code is assigned to each event that can occur on the RSM. The list of events and the codes associated with each event is listed in Appendix D, “OEM Sensor Events”.

Setting event action scripts can be done using any of the standard RSM interfaces (CLI, SNMP, ShM API). The format for the CLI command is as follows:

```
cmmset -l <location> -t <sensor_name> -d eventaction -v
[<time>:]<event_code>:<script> [args]
```

- *event_code* is supplied using either hexadecimal or decimal notation. If hexadecimal notation is used, it must begin with the characters 0x followed by the hexadecimal digits, such as 0x04F8.
- *time* is maximum execution time. If not specified, the default value is used (unlimited time).

This setting is written to the /etc/cmm/policy.conf file and is synched to the standby RSM. It is persistent across boots.

20.2.3 Script Execution

Even though the process of associating scripts can take place only on the active RSM, the scripts can be launched either on the active or on the standby RSM (or on both) depending on where the action that causes the script to be launched occurs.

Caution: The RSM may launch at most one script on a particular event. In certain circumstances, a script can be launched twice on the same event. In particular, in case of failover, a script that did not complete execution on active RSM before failover occurs is relaunched on the new active RSM during failover recovery (this is true for all sensors except for local RSM sensors listed in Table 75, “RSM sensors available on physical address, LUN 02” on page 207). Scripts should be defined in such way that repeated execution does not have a negative effect on the chassis.

A script does not automatically stop running when a sensor returns to a normal setting (no alarms or events). If appropriate, a script must be created to be run when a sensor returns to normal and associate it with that sensor and the action type NormalAction.

Caution: The execution of scripts triggered by health events is monitored. Any script that executes longer than a configured execution time is terminated in a forcible manner (to ensure backward compatibility the default value is unlimited time).

20.2.4 Listing Scripts Associated with Events

To view the script associated with a specific health event for a particular sensor, execute the following command:

```
cmmget –l <location> –t <target> –d <action_type>
```

location is the component in the chassis that the health event is associated with.

target is the sensor that is triggered on.

action_type is `NormalAction`, `MinorAction`, `MajorAction`, or `CriticalAction` depending on the severity of the health event that has been triggered.

To view the scripts associated with specific event codes, view the `/etc/cmm/policy.conf` file and locate the association for the given sensor and event code.

20.2.5 Disassociating Scripts from an Event

To prevent a script from executing when an event on a particular target with which it has been associated occurs, execute the following command:

```
cmmset –l <location> –t <target> –d <action_type> –v none
```

location is the component in the chassis that the health event is associated with.

target is the sensor that triggers the event.

action_type is `NormalAction`, `MinorAction`, `MajorAction`, or `CriticalAction` depending on the severity of the event triggered.

You can verify that no script is associated by entering the `cmmget` command and seeing a blank line as the returned output. For example:

```
cmmget –l blade4 -t "0:Ambient Temp" –d MajorAction
```

This command returns a blank line if no script is associated with the specified event.

To prevent a script from executing after it has been associated with an event, execute the following command:

```
cmmset –l <location> –t <target> –d EventAction –v <event_code>:none
```

20.2.6 Script Synchronization

Scripts stored on the RSM in the `/usr/share/cmm/scripts` directory are synchronized to the standby. Automatic script synchronization occurs:

- as a part of initial synchronization
- upon association of a script to an event

In addition, scripts can be synchronized on user request after editorial changes. Using the `touch` command on the scripts directory has no direct effect on script synchronization. Instead, the CLI provides a command to attain this goal. To force script synchronization, execute the command:

```
cmmset -l cmm -d SynchronizeScript -v <script_name>
```

Scripts are always synchronized by copying scripts from the active RSM to the standby RSM— never from the standby RSM to the active RSM. All changes or additions to scripts on the standby RSM need to be manually copied to the active RSM.

You should always edit scripts on the active RSM rather than the standby RSM. The synching of files in `/usr/share/cmm/scripts` causes the scripts as written on the active RSM to overwrite the corresponding scripts on the standby RSM. Any edits made only on the standby RSM would be lost after a synchronization.

Scripts located in directories outside `/usr/share/cmm/scripts` on the active RSM are *not* synched. These need to be loaded manually onto the standby RSM. Scripts located in those other directories must also be synchronized manually. In other words, any changes made to a script located in one of those other directories on one RSM must be made manually to the corresponding script on the other RSM.

Scripts need to be deleted from both RSMs manually. Deleting a script on the active RSM does not automatically delete the script on the standby when synchronization occurs.

20.3 Environment Variables

Event data is made available through environment variables just prior to the launch of the action script. These environment variables are inherited by the new script, which can inspect the value of these variables as part of its decision logic.

Note: The existence of these environment variables does not affect scripts written to work with previous versions of the firmware.

The names of the environment variables and their meanings are described in Table 36.

Table 36. Environment variables containing event data

Name of Variable	Kind of information	Example
SEL_BLADE	Blade number	0x13
SEL_EVENT_CODE	Event code (See the RSM Software Technical Product Specification for a list of these)	0x0420
SEL_DESCRIPTION	Event description string	Initial Data Synchronization Complete : Assertion, Event Code : 0x0420
SEL_SENSOR_TYPE	Sensor type	0xDE
SEL_SENSOR_NUMBER	Sensor number of the entity	0xE7
SEL_EVENT_DIRECTION	If assertion, then 0. If deassertion, then 1.	1
SEL_EVENT_TYPE	1 for threshold event 2-xx for generic discrete event 6F for sensor specific-specific event	0x6F

Table 36. Environment variables containing event data (Continued)

Name of Variable	Kind of information	Example
SEL_EVENT_DATA_1	ED1	0x03
SEL_EVENT_DATA_2	ED2	0xFF
SEL_EVENT_DATA_3	ED3	0xFF

20.4 Error Processing and Messages

This section describes the error processing performed when associating a script with an event.

Errors are reported in the `/var/log/cmm/error.log` file. The same error message is recorded in the log file regardless of the interface used (CLI, SNMP, or RPC). However, the precise error information returned directly through the invoked interface (CLI, SNMP, or RPC) will vary to some extent depending on the interface used.

The error information returned through the CLI is documented in the rest of this section.

The error information returned when *setting* a value using SNMP consists of the string “BadValue”. The error information returned when *getting* a value using SNMP consists of a string containing the substring “Action Scripts:”. Since this substring will not appear unless an error condition occurs, the output string from the `snmpget` command can be parsed to determine if the substring appears; if it does, an error has occurred.

In RPC the error code is returned in the return packet along with a string that describes the error.

If an error occurs, existing associations of action scripts to events are not modified.

Note: Errors related to action scripts do not contribute to the overall health count of the RSM.

20.4.1 Invalid pathname

If you attempt to associate a script with an absolute pathname that does not begin with `/usr/share/cmm/scripts`, the following error message displays:

`Action Scripts: Invalid Directory` *`directory_name`* `Error. No Association has been made.`

20.4.2 Script does not exist

Attempting to associate a script that does not exist, has a different file name, or is stored in a directory other than the one specified in the `cmmset` command, generates the following error message:

`Action Scripts: File` *`pathname_specified`* `Not Found Error. No association has been made.`

This same message is logged in `error.log` if this check fails when the RSM attempts to execute the script in response to the triggering event.

20.4.3 Pathname specified is a directory

Attempting to associate a directory instead of a file results in the following error message:

`Action Scripts: Associating a Directory (i.e.` *`pathname_specified`*`) is Not Allowed Error. No association has been made.`

20.4.4 Moved or removed script still associated with event

An error occurs if an attempt is made to retrieve the pathname of a script that was associated with an event and where the script was later either deleted or moved without unassociating the script from the event. For example, if a script is associated with a critical action event for the +3.3V target, the pathname of that script is retrieved with the following command:

```
cmmget -t "0:+3.3V" -d CriticalAction
```

If the script is then deleted or moved without unassociating it from the event, the following error message occurs in response to the above command:

Action Scripts: Script *pathname_of_script* Has Been Removed Error. No Association has been made.

This same message is logged in `error.log` if this check fails when the RSM attempts to execute the script in response to the triggering event.

20.4.5 Script has zero bytes

If you attempt to associate a script containing zero bytes, you get the following error message:

Action Scripts: Script *pathname_of_script* is Zero (0) Size Error. No Association has been made.

This same message is logged in `error.log` if this check fails when the RSM attempts to execute the script in response to the triggering event.

20.4.6 Script lacks execute permission

If you attempt to associate a script that does not have execute permission for the owner, you get the following error message:

Action Scripts: Script *pathname_of_script*: No Owner Execute Permissions Error. No Association has been made.

This same message is logged in `error.log` if this check fails when the RSM attempts to execute the script in response to the triggering event.

20.4.7 Script is on the standby RSM

If you attempt to associate a script on the standby RSM to an event, you get the following error message:

```
cmmset: This is the standby CMM. Please execute this operation on the active CMM.
```

The active CMM's IP addresses are *ip_address* and *ip_address*.

20.4.8 Unable to write to policy.conf

Associations between scripts and events are recorded in the `/etc/cmm/policy.conf` file. If the RSM is unable to write to this file, an error is reported.

20.5 Default Scripts

Radisys ships the RSM with a number of default scripts located in the `/usr/share/cmm/scripts` directory. In addition, the `/etc/cmm/policy.conf` file contains a set of event-to-script associations that trigger event scripting for default scripts.

20.6 Limitations

This section describes some assumptions and limitations that pertain to RSM scripting.

20.6.1 Usage of switchover commands

In order to prevent ping-pong behavior, user scripts calling switchover or failover CLI commands defined in section Chapter 10.0, “High Availability” on page 49 must adhere to the following limitations:

- The script calling the switchover command can only be associated with events from sensors exposed by the RSM at HW address, LUN 02. Refer to Appendix A, “RSM Sensors - Physical IPMC” on page 205 for a list of such sensors.
- The switchover command is called as the last command in the script.

Chapter 21

21.0 Operational State Management

A FRU enters an AdvancedTCA* shelf and goes through a series of hot swap states to become active. Likewise, a FRU transitions through a series of hot swap states as it deactivates in preparation for extraction from the AdvancedTCA* shelf. The IPMC maintains the hot swap state for the FRU and additional sub-FRUs present on the FRU, and emits an event for each state transition.

The RSM manages FRU insertions, extractions, and the operational states and state transitions of the nodes in a shelf in accordance to Section 3.2.4 of "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". For each FRU, it handles received hot swap events, tracks the current state of the FRU, and sends requests to change the FRU hot swap state.

21.1 Hot Swap States

Hot swap states and transitions are defined in "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". These states are:

- M0 - Not Installed
- M1 - Inactive
- M2 - Activation Request
- M3 - Activation In Progress
- M4 - Active
- M5 - Deactivation Request
- M6 - Deactivation In Progress
- M7 - Communication Lost

The RSM caches the hot swap state for each FRU. To get the hot swap state of a FRU cached by the RSM, execute the command:

```
cmmget -l <location> -d HotSwapState
```

where <location> stands for a valid location (i.e. FRU name) as defined in "Alert Standard Format (ASF) Specification version 2.0".

21.2 Hot Swap Sensor

Each IPMC hosts one "Hot Swap" Sensor for each FRU that it represents. The "Hot Swap" sensor indicates the current hot swap state, previous state, and the cause of the state transition. For a detailed description, refer to Appendix D, "OEM Sensor Events".

To retrieve the current hot swap state for location (as opposed to the value most recently cached by the RSM), query the current value of the "Hot Swap" sensor for location directly:

```
cmmget -l <location> -t "Hot Swap" -d current
```

where "Hot Swap" is the name of the Hot Swap sensor on the indicated location. For a detailed description, refer to Appendix D, "OEM Sensor Events".

21.3 FRU Control Scripts

The RSM ships with these default FRU control scripts located in the `/usr/share/cmm/scripts` directory:

- FRU activate script
- FRU deactivate script

A FRU hot-swap state change from M1 to M2 causes the generation of a hot-swap event by the IPMC, which, when processed by the RSM, triggers the FRU activate script. The script checks the "Shelf Manager Controlled Activation" bit in the FRU Activation and Power Management Record for that FRU. If the bit is set to 0 (system manager activates FRU), the scripts exits. If the bit is set to 1 (shelf manager activates FRU), the script performs activation using this CLI command:

```
cmmset -l <location> -d FruActivation -v 1
```

A FRU hot-swap state change from M4 to M5 causes the generation of a hot-swap event by the IPMC, which, when processed on the RSM, triggers the FRU deactivate script. The default script performs deactivation using this CLI command:

```
cmmset -l <location> -d FruActivation -v 0
```

The above description addresses all locations except RSMs. The activation and deactivation of the RSM itself is not controlled by the FRU control script.

21.4 FRU Activation Policy

The current FRU Activation Policy can be set with this command:

```
cmmset -l <location> -d FruActivationPolicy -v {0|1}
```

To query the current FRU Activation Policy, execute this command:

```
cmmget -l <location> -d FruActivationPolicy
```

A matching dataitem `FruDeactivationPolicy` is used to set/get the FRU De-activation Policy.

21.5 Checking Node Presence

The RSM periodically verifies the presence of each node in the shelf and alerts the System Manager when it loses contact with it. The following table lists configuration parameters stored in `shm.conf` for time delay and the number of pings that the RSM uses to determine the state of a FRU.

Table 37. Ping configuration

Variable	Description	Value
CLD_PING_INTERVAL	Minimum time between consecutive pings of the same FRU [ms].	6000
CLD_PINGS_PER_SEC	Maximum number of pings per second (HW limitation) [1/s].	10
CLD_MAX_FAILED_PINGS	How many failed attempts to contact the IPMC must occur prior to raising an event that communication has been lost.	2

The actual delay between two consecutive pings is calculated from the formula:

PingDelay = max{CLD_PING_INTERVAL/NumberIPMCs, 1/CLD_PINGS_PER_SEC}.

22.0 Power Management

The RSM controls power to the nodes of a chassis. The RSM grants power to each FRU after negotiating with the respective IPMI device fronting the FRU. The RSM also manages the power budget of each power feed. The RSM uses shelf FRU information to guarantee power-up sequence and delays between boards and to ensure that maximum FRU power capability is not violated.

Upon user request the RSM can power up, power down, and reset a blade in a particular slot and can be used to query the power state of a blade at any time.

With two RSMs operating in redundant mode the active RSM is responsible for power management. Critical power management data is kept in sync at all times between the active and standby RSMs. The standby RSM does not participate in any power management activities.

22.1 Node Operational Power Management

The RSM manages power negotiations, allocation and reclaim for all nodes in a shelf in accordance to Section 3.9 of "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification".

The "Power Allocation" Sensor on the RSM tracks the power negotiation process. Refer to Appendix D, "OEM Sensor Events" for a detailed sensor definition.

When a FRU is discovered in M7 state, the RSM needs to reserve power for that FRU. A configuration parameter `POWER_UNKNOWN_FRU` specifies the amount of power reserved in this case.

Table 38. Power configuration

Variable	Description	Value
POWER_UNKNOWN_FRU	Indicates the power budget that will be reserved for each FRU that is discovered in M7 state [0.1W]	2000

22.1.1 Power Levels

The RSM can be queried for the supported power levels of each node using this CLI command:

```
cmmget -l <location> -d PowerLevels
```

To display the currently assigned power level, execute the command:

```
cmmget -l <location> -d PresentPowerLevel
```

22.1.2 Shelf Power Budget

The RSM can show the current shelf power budget with this CLI command:

```
cmmget -d PowerBudget
```

Alternatively, you can query the "Power Budget" Sensor on RSM location. Refer to Appendix D, "OEM Sensor Events" for a detailed sensor definition.

22.1.3 Power-on Sequence

The power-on sequence is determined by the order of Power Descriptor entries in the Shelf Activation and Power Management Record in the Shelf FRU "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification".

To get the power-on sequence, execute the command:

```
cmmget -d PowerSequence
```

The RSM does not support the `cmmset` command for the `PowerSequence` dataitem. Changes to the power-on sequence must be made using the FRU update utility described in Chapter 34.0, “FRU Update Utility” on page 176.

22.2 Power Feed Targets

The CLI allows certain `cmmget` queries to be taken on power feeds for a location. They include the following dataitems: `maxExternalAvailableCurrent`, `maxInternalCurrent`, and `minExpectedOperatingVoltage`. These dataitems are described in “Alert Standard Format (ASF) Specification version 2.0”.

To find the number of feed targets, execute this command:

```
cmmget -d FeedCount
```

This returns an integer indicating the number of power feeds.

For example, the RSM installed in the MPCHC0001 chassis returns the number 4 in response to the above command. The MPCHC0001 chassis has four power feeds coming from the PEMs: feed1, feed2, feed3, and feed4. These correlate to the physical feeds on the MPCHC0001 as follows:

feed1 = FeedA1
feed2 = FeedB2
feed3 = FeedA2
feed4 = FeedB1

Refer to the documentation for your chassis for more information on the power feeds.

22.3 Forced Power State Changes on Blades

You can request power state changes for blades, such as power on, power off, or reset. The RSM is responsible for handling these requests.

22.3.1 Powering Off a Blade

The following command powers off a blade:

```
cmmset -l <bladen> -d PowerState -v poweroff
```

This command sends the PICMG 3.0 Set Fru Activation(Deactivate FRU).

n is the number of the physical slot in which the blade to be powered off is inserted. You are prompted to enter “y” (for “yes”) to confirm that the blade should be powered off before the command actually powers off the blade.

"PowerOff" is not supported on the RSM location.

22.3.2 Powering On a Blade

The following command powers on a blade:

```
cmmset -l <bladen> -d PowerState -v poweron
```

This command sends the PICMG 3.0 Set FRU Activation Policy command to clear the Locked bit. *n* is the number of the physical slot in which the blade to be powered on is inserted.

22.3.3 Resetting a Blade

The following command resets a blade:

```
cmmset -l <blade n> -d PowerState -v reset
```

This command sends the PICMG 3.0 FRU Control command with the Cold Reset option.

n is the number of the physical slot in which the blade to be reset is inserted.

If "reset" is used on RSM location, the software will check for redundancy and a reset will only occur if a redundant peer is identified.

Note: You are prompted to enter “y” (for “yes”) to confirm that the blade should be reset before the command actually resets the blade.

22.4 Obtaining the Power State of a Blade

To obtain the power state information of a blade at any time, execute the following command:

```
cmmget -l <blade n> -d PowerState
```

n is the number of the physical slot in which the queried blade is inserted. This command provides information on whether the blade is present, the power state, and the hot swap state.

Chapter 23

23.0 Cooling and Fan Control

The RSM controls chassis cooling and fan tray settings in accordance with Section 3.9 of the "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". In discovery stage, the RSM queries fan trays for cooling capabilities. In normal operation stage, the RSM monitors temperature events occurring in the chassis.

Thermal conditions in the chassis may change due to fan failure or a clogged filter. Boards that exhibit temperature conditions raise temperature events. When a temperature event is asserted, the RSM adjusts the fan level to adapt to the changing conditions of the chassis or the surrounding environment.

23.1 Temperature Condition Sensor

The "Temperature Condition" Sensor tracks all asserted temperature events in the chassis.

The four temperature levels are:

- Normal – There is currently no asserted temperature event.
- Minor – There is at least one asserted minor temperature event.
- Major – There is at least one asserted major temperature event.
- Critical – There is at least one asserted critical temperature event.

To read the current temperature level, execute the following command:

```
cmmget –d temperaturelevel
```

Alternatively, the sensor can be queried directly. Refer to Appendix D, "OEM Sensor Events" for detailed sensor definition.

23.2 Cooling Policy

The RSM does not use a cooling table to control chassis cooling. Instead, the RSM uses a cooling policy for this purpose.

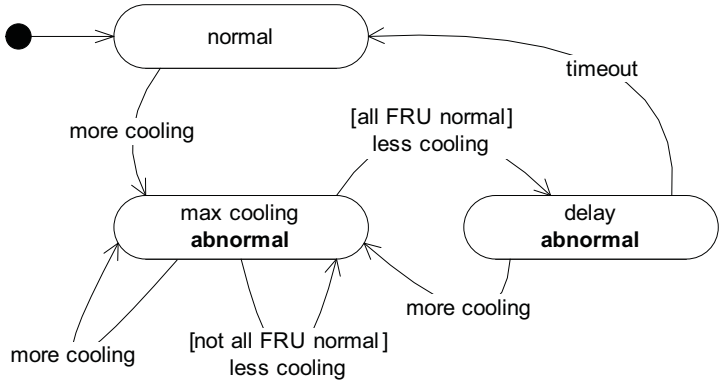
The RSM cooling policy implements cooling level adjustments in accordance with "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". The policy increases fan levels to maximum levels when an abnormal temperature conditions are detected in the shelf, and restores fan levels to normal levels when temperature conditions return to normal.

The cooling policy is always in one of three states. The states reflect current cooling levels forced by the policy.

- Normal - represents the state in which all fan levels are set to normal level. No temperature event is asserted.
- Abnormal - represents the state in which fan levels are set to maximum level due to existing asserted temperature events or during re-enumeration.
- Delay - represents the state in which fan levels are temporarily left at maximum level to extend the time until policy returns to normal.

The RSM implements the "Cooling Policy" sensor, which tracks cooling policy states. For a detailed description, refer to Appendix D, "OEM Sensor Events".

Figure 4. Cooling Policy State Transitions



When the RSM cooling policy receives a request to increase cooling, it sets all fans to maximum speed if the policy is in the 'normal' state. If the request is received in 'delay' state, the scheduled timer is canceled. The cooling policy changes its state to 'abnormal'.

When the RSM cooling policy receives a request to decrease cooling, it first checks conditions on all FRUs. If all FRUs are restored to 'normal' state, the cooling policy starts a delay timer. This timer is used to delay the fan level restoration procedure and prevent the cooling policy from oscillating between Normal and Abnormal as the temperature runs along just above and below the threshold value.

The initial delay value is equal to the value of the `COOLING_DELAY_STEP` parameter stored in the `/etc/cmm/shm.conf` configuration file. The subsequent values are calculated from the previous values `+/-` the value of the `COOLING_DELAY_STEP` parameter, depending on how long the cooling policy has stayed in Normal state.

When a delay timer expires, the RSM cooling policy restores all fan levels to normal and changes its state to 'normal'. The cooling policy stores the current time to allow timer delay modifications in case of repeated abnormal condition re-occurrences within a short time of restoring normal fan levels.

When a critical shelf-related temperature event is detected, the cooling policy begins to power off individual FRUs. This behavior is configurable through the configuration parameter `COOLING_IGNORE_CRITICAL_TEMP_SHELF` (disabled by default), and can be switched on or off subject to system manager requirements.

The value of the `COOLING_DEACTIVATION_STEP` parameter is used to determine how long to wait between powering off FRUs.

Similarly, when a critical temperature event from a blade is detected, the cooling policy powers off the FRU. Again, this behavior is a configurable feature controlled by configuration parameter `COOLING_IGNORE_CRITICAL_TEMP_FRU` (enabled by default), and can be switched on or off subject to system manager requirements.

The `POWERON_IGNORE_CRITICAL_TEMP_SHELF` parameter configures the cooling policy behavior so FRUs are powered on if a critical shelf temperature condition is present. Setting the parameter value to `1` enables this behavior. No failover occurs, so the active RSM powers on the FRU. The default value for this parameter is `0`, which specifies the FRUs will not be powered on if a critical shelf-related temperature event exists.

All of these cooling policy parameters are stored in the `/etc/cmm/shm.conf` configuration file. See Table 39 on page 117 for more information about the cooling policy parameters.

Caution: Some blades may not support critical temperature events. To handle such blades safely, the user may associate a user script with major temperature events from such blades. The script must send a power off request to the blade in a proactive manner if configuration parameter `COOLING_IGNORE_CRITICAL_TEMP_FRU` is set to zero.

Table 39. Cooling Configuration

Variable	Description	Value
COOLING_DELAY_STEP	Cooling delay step is used to set the initial delay value of cooling policy [ms]	10000
COOLING_DEACTIVATION_STEP	Cooling deactivation step is used to determine how long to wait between powering off individual FRUs when a critical, shelf related, temperature event is detected [ms]	5000
COOLING_IGNORE_CRITICAL_TEMP_SHELF	Logical flag used to determine whether cooling policy must power off individual FRUs upon shelf related critical temperature event.	1
COOLING_IGNORE_CRITICAL_TEMP_FRU	Logical flag used to determine whether cooling policy must power off the FRU upon FRU related critical temperature event.	0
POWERON_IGNORE_CRITICAL_TEMP_SHELF	Logical flag used to determine whether cooling policy must power on the FRU upon shelf-related critical temperature event.	0

23.2.1 Process for modifying the shm.conf file

The `/etc/cmm/shm.conf` file contains a list of the RSM cooling policy parameters and their values. Changes to the cooling policy are accomplished by modifying the parameter values in `shm.conf`.

Changes to `shm.conf` should be done after stopping the `cmm` service. The updated `shm.conf` file is then synchronized to the standby RSM during RSM startup. Follow these steps:

1. Stop the `cmm` service in both RSMs.
   `cmm stop`
2. Modify the `shm.conf` file in one of the RSMs (either RSM1 or RSM2).
3. Start the RSM with the modified file.
   `cmm start`
4. When the RSM becomes Active No Standby, start the other RSM so the file changes are synchronized to the standby RSM.

Alternative steps

1. Stop the `cmm` service in both RSMs.
   `cmm stop`
2. Modify the `shm.conf` file in both RSMs.
3. Start the `cmm` service in both RSMs.
   `cmm start`

23.2.2 Normal Cooling Adjustments

The RSM cooling policy does not support cooling adjustments under normal operating conditions. After fan levels are restored to normal (maximum sustained level), no further fan level optimizations are performed.

Normal cooling adjustments can be performed by means of user scripts associated with the "Cooling Policy" sensor events. These scripts can be customized to a specific shelf and use selected events to trigger fan level modifications over CLI.

Caution: Abnormal temperature events generated as a result of improper script actions will trigger the RSM to take corrective action.

23.3 Fan Control in Re-enumeration

At the start of chassis re-enumeration the RSM drives the fans to full speed (100 percent). The speeds are not brought back to normal level until re-enumeration is finished and the RSM has determined that there are no thermal events in the chassis.

23.4 Fan Tray Cooling Properties

The fan tray supports a range of cooling levels at which it operates.When queried via IPMI, the fan tray returns its maximum cooling level, minimum cooling level and a recommended cooling level for normal operation. The AdvancedTCA* specification states that fan trays must support all cooling levels between its minimum and maximum levels by increments of one unit.

The fan tray can run at only one cooling level at a time.

A given cooling levels does not correlate with a certain fan speed because a cooling unit may not actually contain fans. In fact, the RSM is unaware of how the fan trays cool the chassis. It simply knows that to increase the cooling output of the fan tray it should use a higher cooling level. Each fan tray may (and most likely will) have different minimum, maximum and recommended normal cooling levels.

To get the minimum cooling level that the fan tray supports, execute this command:

```
cmmget –l <fantrayn> -d minimumsetting
```

To get the maximum cooling level that the fan tray supports, execute this command:

```
cmmget –l <fantrayn> -d maximumsetting
```

To get the fan tray's recommended cooling level, execute this command:

```
cmmget –l <fantrayn> -d recommendedsetting
```

To get the fan tray properties, execute the command:

```
cmmget –l <fantrayn> -d properties
```

n is the number of the fan tray being addressed.

23.5 Retrieving Current Cooling Level

You can get the current cooling level by executing this command:

```
cmmget –l <fantrayn> –d currentfanlevel
```

n is the number of the fan tray being addressed.

This command queries the fan tray and returns the current cooling level. If the fan tray is in Fantray Control Mode, the cooling level selected by the fan tray is returned. If the fan tray is in emergencyshutdown mode, "0" is returned.

23.6 Setting Current Cooling Level

User scripts performing normal cooling adjustments can change the current cooling level by executing this command:

```
cmmset –l <fantrayn> –d fanlevel -v <fanlevel>
```

n is the number of the fan tray being addressed.

23.7 Fan Tray Sensors

To query the fan tray and fan tray sensors, specify fantray*n* as the location (`-l FanTray`*n*) in the `cmmget` command. For example, to query the current RPM value of a fan in the fan tray 1 on a chassis, execute the command:

```
cmmget -l fantray1 -t "<fan speed sensor name>" -d current
```

The return value might look like this:

```
The current value is 3325.000 RPM
```

23.8 Control Modes for Fan Trays

There are three modes of control that a fan tray may operate at:

- Cmm
- Fantray
- Emergency Shutdown

The `DefaultControl` option is not supported. The fan tray runs at exactly one control mode at a time. The control mode that the fan tray is running at is its current control mode. You can change the current control mode of each fan tray in the shelf.

To get the current control mode, execute the command:

```
cmmget –l <fantrayn> -d control
```

23.8.1 RSM Control Mode

The RSM Control Mode is the mode in which the RSM has complete control over the fan tray's current cooling level. In RSM Control Mode the RSM uses the cooling policy to determine which cooling level to use for the current temperature status.

You can change to this mode with the following command:

```
cmmset –l <fantrayn> -d control –v cmm
```

n is the number of the fan tray being addressed.

23.8.2 Fantray Control Mode

The AdvancedTCA specification defines a mode called local control where the fan tray determines its own cooling level. The control mode can be local mode only if there are no temperature events in the chassis.

The RSM does not support fan tray local control mode.

23.8.3 Emergency Shutdown Control Mode

The Emergency Shutdown control mode causes the fan tray to stop cooling the system. A fan tray stays in this mode until the current control mode is changed to one of the other two modes.

To change to this mode, execute the following command:

```
cmmset –l <fantrayn> -d control –v emergencyshutdown
```

n is the number of the fan tray being addressed.

Note: Not all fan trays support emergency shutdown control mode.

23.9 Automatic Control Mode Change

The fan tray's current control mode can be changed automatically rather than as the result of executing an explicit CLI command. In the case where the fan tray is in Fantray control mode and a temperature event is asserted, the fan tray should not control itself. Instead, the RSM executes the cooling policy and increases the current cooling level. Once this change in control takes place, the fan trays stay in RSM control mode until you specify otherwise. If this automatic change in control mode occurs, a SEL event is logged and an SNMP trap is sent.

23.10 Fan Tray LED

The RSM controls the fan tray LEDs. In a healthy state (no events), the LED is set to display the color green. If any of the fan tray sensors (temperature, voltages, fan tachometers) are in an unhealthy state, the LED is set to display the color red or the color amber. (The color red is displayed by default).

Chapter 24

24.0 Electronic Keying Management

Electronic Keying (EKeying) is used in the AdvancedTCA architecture to dynamically implement a specific fabric interconnect in a fabric agnostic backplane. The PICMG 3.0 Specification calls out two types of EKeying: point-to-point and bused.

24.1 Point-to-Point EKeying

Point-to-point EKeying is used to set up a specific fabric interconnect and protocol between two end points when a board is inserted into the chassis.

With point-to-point EKeying the RSM queries the topology of the interconnects in the shelf from the shelf FRU multi-records, determines each board's EKeys from the Board FRU multi-records, and attempts to find the best match possible between the two interconnected end-points. Once the match is made, the RSM directs each of the entities to enable its interconnect and informs the entities which protocol to use. If no match is found, the two end points are directed to disable their interconnect.

24.2 Bused EKeying

Bused EKeying is used to manage control of the bused resources provided by an AdvancedTCA chassis. These resources include the Synchronization Clock Interface and the Metallic Test Bus.

With bused EKeying the RSM grants control of a specific resource to a single requesting board. Only one board can control a resource at any given time. The RSM controls the resources through the use of tokens. A board can request the token for a particular resource from the RSM at any time. If the RSM has possession of the token for that resource, it grants the token to the requesting board. If the RSM does not have possession of the token, the requesting board is notified and the token owner is notified that it will need to release the token as soon as possible.

24.3 EKeying CLI Commands

The CLI on the RSM includes two dataitems used with the `cmmget` command to obtain EKeying information for the system.

To retrieve the EKeys that have been granted to the board, execute the command:

```
cmmget -l <location> -d grantedboardekeys
```

To retrieve a list of Bused EKeys and learn who owns them, execute the command:

```
cmmget -d busedekeys
```

Refer to "Alert Standard Format (ASF) Specification version 2.0" for more information on these CLI dataitems.

25.0 CDMs, Shelf FRU, and FRU Information

25.1 Chassis Data Modules

There are two chassis data modules (CDMs) in a single chassis to provide high availability and fault tolerance through redundancy. Each CDM has an EEPROM containing the FRU information for the chassis. The CDM stores serial number and asset information about the chassis and provides PICMG 3.0 shelf FRU information, such as the number of slots, slot connection/routing information (for electronic keying), maximum power per feeds, and so on.

There is no direct access to CDM devices at the system management interface level. The two CDM devices are fronted by one instance of shelf FRU information selected during the election process.

Note: The RSM always assumes CDMs are present in the chassis . Do not remove the CDMs once power is applied to the chassis.

25.2 Shelf FRU Election Process

Once started, the RSM needs to elect which CDM's data to use to retrieve critical chassis information. The following two data sets are compared during shelf FRU election:

- CDM1
- CDM2

The RSM creates caches once the shelf FRU election is completed successfully. The shelf FRU election process fails if none of the CDM devices are valid.

Upon failed shelf FRU election the RSM goes to out-of-service state, where corrective steps can be taken to ensure success in the next election.

25.3 Shelf FRU Information

The location `chassis:254` refers to the shelf FRU after the election process is finished. The only target that can be specified with this location is FRU. The following command can be used to retrieve all the shelf FRU information:

```
cmmget -l chassis:254 -t FRU -d all
```

Other dataitems can be used to retrieve specific fields of data in the shelf FRU. To see what those dataitems are, execute this command:

```
cmmget -l chassis:254 -t FRU -d listdataitems
```

25.4 FRU Information

The RSM can query the entire FRU of a device, entire areas of a FRU, or individual fields in the different areas of the FRU. The set of supported dataitems matches the FRU information storage layout as defined in "Platform Management FRU Information Storage Definition".

FRU information is stored in non-volatile memory and is used by the IPMC to locate and communicate with the available FRUs.

25.4.1 Physical IPMC FRU 0

The IPMC uses 1KB of the SPI flash for the physical IPMC FRU 0 information storage. The overall FRU 0 information organization is described in the following table.

Table 40. Dataitems Used With FRU Target to Obtain FRU Information

FRU Area	Size (in bytes)
Header	8
Internal area	0
Chassis	0
Board information area	*calculated
Product information area	*calculated
Multi-record area	*calculated
Total size	1024

25.4.1.1 Header

The FRU information header contains the version of the FRU storage format specification and offsets to the various sections of the FRU information.

25.4.1.2 Internal Area

The internal area is a private, non-volatile storage area allocated to the IPMC for implementation-specific purposes. The area is not used, so its size is 0.

25.4.1.3 Board Information Area

The board information area contains information about the board where the FRU information device is located. The following table lists the field descriptions and values.

Table 41. Physical IPMC FRU 0 Board information area (Sheet 1 of 2)

Field Description	Size (in bytes)	Default Value (hex)
Format Version	1	0x01
Board Area Length	1	*calculated
Language Code	1	0x19 - English
Manufacturer Date/Time	3	*based on manufacturing data
Board Manufacturer type/length	1	0xCD
Board Manufacturer	13	Radisys Corp.
Board Product Name type/length	1	0xD4
Board Product Name bytes	20	A6K-RSM-J *padded at the end with spaces
Board Serial Number type/length	1	0xCD
Board Serial Number	13	*programmed by manufacturing
Board Part Number type/length	1	0xD4
Board Part Number	20	*programmed by manufacturing
FRU File ID type length	1	0xC0
Board Custom 1 type/length	1	0xD4
Board Custom 1	20	*customer specific
Board Custom 2 type/length	1	0xD4
Board Custom 2	20	*customer specific

Table 41. Physical IPMC FRU 0 Board information area (Sheet 2 of 2)

Field Description	Size (in bytes)	Default Value (hex)
Board Custom 3 type/length	1	0xD4
Board Custom 3	20	*customer specific
No more fields	1	0xC1
Padding	*calculated	0x00
Board Area Checksum	1	*calculated
Total size	*calculated	

25.4.1.4 Product Information Area

The product information area contains information about the FRU itself.

Table 42. Physical IPMC FRU 0 Product information area

Field Description	Size (in bytes)	Default Value (hex)
Format Version	1	0x01
Product Area Length	1	*calculated
Language Code	1	0x19 – English
Manufacturer Name type/length	1	0xCD
Manufacturer Name	13	Radisys Corp.
Product Name type/length	1	0xC9
Product Name	9	A6K-RSM-J
Product Part/Model Number type/length	1	0xCE
Product Part/Model Number	14	*programmed by manufacturing
Product Version type/length	1	0xD4
Product Version	20	*spaces
Product Serial Number type/length	1	0xCD
Product Serial Number	13	*programmed by manufacturing
Asset Tag type/length	1	0xD4
Asset Tag	20	*customer specific
FRU File ID type length	1	0xC5
FRU File ID	5	XX.YY (FRU template version) *not changed during mfg
Product Custom 1 type/length	1	0xD4
Product Custom 1	20	*customer specific
Product Custom 2 type/length	1	0xD4
Product Custom 2	20	*customer specific
Product Custom 3 type/length	1	0xD4
Product Custom 3	20	*customer specific
End of Fields	1	0xC1
Padding	*calculated	0x00
Product Area Checksum	1	*calculated
Total size	*calculated	

25.4.1.5 Multi-record Area

The multi-record area contains records about shelf management and E-Keying configurations.

25.4.1.5.1 Radisys Shelf Management Configuration Record

This record configures the shelf manager functionality of the IPMC. It can disable shelf management, or enable it in basic mode or enhanced mode. Enhanced mode runs the full ATCA shelf manager compliant with the ATCA specification, while basic mode is a simple shell script to power up a shelf. The record also configures the redundant addresses where the IPMC should power up as a shelf manager.

Table 43. Multi-record area: Shelf management configuration record

Field Description	Size (in bytes)	Default Value (hex)
Record Type ID	1	0xC0
End of List/Version	1	0x02
Record Length	1	0x08
Record Checksum	1	*calculated
Header Checksum	1	*calculated
Manufacturer ID (LS byte first)	3	0xF1 0x10 0x00
PICMG Record ID	1	0x09
Record Format Version	1	0x01
Shelf Management Enable & Mode	1	0x01 ATCA shelf manager enabled
Redundant Address 1	1	0x10
Redundant Address 2	1	0x12
Total size	*calculated	

25.4.1.5.2 PICMG Board Point to Point Connectivity Record

This record contains the E-Keying information for establishing interface connections on the ATCA backplane. Refer to Electronic Keying under the Hardware Platform Management section of the ATCA specification for details about how these values are derived.

Table 44. Multi-record area: PICMG board point to point connectivity record

Field Description	Size (in bytes)	Default Value (hex)
Record Type ID	1	0xC0
End of List/Version	1	0x82
Record Length	1	*calculated
Record Checksum	1	*calculated
Header Checksum	1	*calculated
Manufacturer ID (LS byte first)	3	0x5A 0x31 0x00
PICMG Record ID	1	0x14
Record Format Version	1	0x00
OEM GUID Count	1	0x00
OEM GUID	0	
Link Descriptors (LS byte first)	N * 4	See Table 45
Total size	*calculated	

Link descriptors include those for base interface shelf manager cross connect and standard PICMG 3.0 10/100/1000 links. Table 45 describes the link descriptors in detail.

Table 45. Link descriptors

Port	Bits: 31:24 Grouping ID	Bits: 23:20 Type Ext	Bits: 19:12 Link Type	Bits: 11:0 Link Designator	Descriptor
Base Channel 1 ShMC X-connect	0000 0000'b	0001'b	0000 0001'b	0001 0000 0001'b	0x00101101
Base Channel 2 ShMC X-connect	0000 0000'b	0001'b	0000 0001'b	0001 0000 0010'b	0x00101102
Base Channel 1 PICMG 3.0	0000 0000'b	0000'b	0000 0001'b	0001 0000 0001'b	0x00001101
Base Channel 2 PICMG 3.0	0000 0000'b	0000'b	0000 0001'b	0001 0000 0010'b	0x00001102

25.4.1.5.3 PICMG LED Description Record

This record contains information about the main FRU LEDs. Refer to LED Description Record under the Hardware Platform Management section of the ATCA specification for details about how these values are derived.

Table 46. Multi-record area: PICMG LED description record (Sheet 1 of 2)

Field Description	Size (in bytes)	Default Value (hex)
Record Type ID	1	0xC0
End of List/Version	1	0x82
Record Length	1	*calculated
Record Checksum	1	*calculated
Header Checksum	1	*calculated
Manufacturer ID (LS byte first)	3	0x5A 0x31 0x00
PICMG Record ID	1	0x2F
Record Format Version	1	0x00
LED Descriptor Count	1	0x04
ATCA LED 0 descriptor		
LED ID	1	0x00 - Blue LED
LED Legend Type/Length Byte	1	0xC2
LED Legend	2	"HS"
LED Symbol Type/Length Byte	1	0xC0
LED Symbol	0	
LED Description Type/Length Byte	1	0xC0
LED Description	0	
ATCA LED 1 descriptor		
LED ID	1	0x01 - OOS LED
LED Legend Type/Length Byte	1	0xC3
LED Legend	2	"OOS"
LED Symbol Type/Length Byte	1	0xC0
LED Symbol	0	
LED Description Type/Length Byte	1	0xC0
LED Description	0	
ATCA LED 2 descriptor		
LED ID	1	0x02 - PWR LED
LED Legend Type/Length Byte	1	0xC3

Table 46. Multi-record area: PICMG LED description record (Sheet 2 of 2)

Field Description	Size (in bytes)	Default Value (hex)
LED Legend	2	"PWR"
LED Symbol Type/Length Byte	1	0xC0
LED Symbol	0	
LED Description Type/Length Byte	1	0xC0
LED Description	0	
ATCA LED 3 descriptor		
LED ID	1	0x03 - ACT LED
LED Legend Type/Length Byte	1	0xC3
LED Legend	2	"ACT"
LED Symbol Type/Length Byte	1	0xC0
LED Symbol	0	
LED Description Type/Length Byte	1	0xC0
LED Description	0	
Total size	*calculated	

25.4.2 Virtual IPMC FRU 0

The IPMC uses 1KB of the SPI flash for the virtual IPMC FRU 0 information storage. The overall FRU 0 information organization is described in the following table.

Table 47. Virtual IPMC FRU 0 Information Summary

FRU Area	Size (in bytes)
Header	8
Internal area	0
Chassis	0
Board information area	*calculated
Product information area	*calculated
Multi-record area	0
Total size	1024

25.4.2.1 Header

The FRU information header contains the version of the FRU storage format specification and offsets to the various sections of the FRU information.

25.4.2.2 Internal Area

The internal area is a private, non-volatile storage area allocated to the IPMC for implementation-specific purposes. The area is not used, so its size is 0.

25.4.2.3 Board Information Area

The board information area contains information about the board where the FRU information device is located. The following table lists the field descriptions and their related data.

Table 48. Virtual IPMC FRU 0 Board information area

Field Description	Size (in bytes)	Default Value (hex)
Format Version	1	0x01
Board Area Length	1	*calculated
Language code	1	0x19 – English
Manufacturer Date/Time	3	*based on mfg. date
Board Manufacturer type/length	1	0xCD
Board Manufacturer	13	Radisys Corp.
Board Product Name type/length	1	0xD4
Board Product Name bytes	20	VFRU-A6K-RSM-J *padded at the end with spaces
Board Serial Number type/length	1	0xCD
Board Serial Number	13	*programmed by manufacturing
Board Part Number type/length	1	0xD4
Board Part Number	20	*programmed by manufacturing
FRU File ID type/length	1	0xC0
Board Custom 1 type/length	1	0xD4
Board Custom 1	20	*customer specific
Board Custom 2 type/length	1	0xD4
Board Custom 2	20	*customer specific
Board Custom 3 type/length	1	0xD4
Board Custom 3	20	*customer specific
No more fields	1	0xC1
Padding	*calculated	0x00
Board Area Checksum	1	*calculated
Total size	*calculated	

25.4.2.4 Product Information Area

The product information area contains information about the FRU itself.

Table 49. Virtual IPMC FRU 0 Product information area (Sheet 1 of 2)

Field Description	Size (in bytes)	Default Value (hex)
Format Version	1	0x01
Product Area Length	1	*calculated
Language Code	1	0x19 – English
Manufacturer Name type/length	1	0xCD
Manufacturer Name	13	Radisys Corp.
Product Name type/length	1	0xCE
Product Name	14	VFRU-A6K-RSM-J
Product Part/Model Number type/length	1	0xCE
Product Part/Model Number	14	*programmed by manufacturing

Table 49. Virtual IPMC FRU 0 Product information area (Sheet 2 of 2)

Field Description	Size (in bytes)	Default Value (hex)
Product Version type/length	1	0xD4
Product Version	20	*spaces
Product Serial Number type/length	1	0xCD
Product Serial Number	13	*programmed by manufacturing
Asset Tag type/length	1	0xD4
Asset Tag	20	*customer specific
FRU File ID type length	1	0xC5
FRU File ID	5	XX.YY (FRU template version) *not changed during mfg
Product Custom 1 type/length	1	0xD4
Product Custom 1	20	*customer specific
Product Custom 2 type/length	1	0xD4
Product Custom 2	20	*customer specific
Product Custom 3 type/length	1	0xD4
Product Custom 3	20	*customer specific
End of Fields	1	0xC1
Padding	*calculated	0x00
Product Area Checksum	1	*calculated
Total size	*calculated	

25.4.3 Virtual IPMC FRU 1

FRU 1 of the virtual IPMC provides methods for accessing the first shelf FRU data device. The format of the FRU information is defined by the shelf implementation.

25.4.4 Virtual IPMC FRU 2

FRU 2 of the virtual IPMC provides methods for accessing the second shelf FRU data device. The format of the FRU information is defined by the shelf implementation.

25.4.5 Virtual IPMC FRU 3

FRU 3 of the virtual IPMC provides methods for accessing the Shelf Alarm Panel (SAP) FRU data device. The format of the FRU information is defined by the SAP implementation.

25.4.6 Virtual IPMC FRU 4

FRU 4 of the virtual IPMC provides methods for accessing the fan tray 1 FRU data device. The format of the FRU information is defined by the fan tray implementation.

25.4.7 Virtual IPMC FRU 5

FRU 5 of the virtual IPMC provides methods for accessing the fan tray 2 FRU data device. The format of the FRU information is defined by the fan tray implementation.

25.4.8 Virtual IPMC FRU 6

FRU 6 of the virtual IPMC provides methods for accessing the fan tray 3 FRU data device. The format of the FRU information is defined by the fan tray implementation.

This FRU is not present when the RSM is installed in a two-slot shelf, since there are only two fan trays.

25.4.9 Virtual IPMC FRU 7

FRU 7 of the virtual IPMC provides methods for accessing the PEM A FRU data device. The format of the FRU information is defined by the PEM implementation.

This FRU is not present when the RSM is installed in a two-slot shelf, since the PEMs are not field replaceable units.

25.4.10 Virtual IPMC FRU 8

FRU 8 of the virtual IPMC provides methods for accessing the PEM B FRU data device. The format of the FRU information is defined by the PEM implementation.

This FRU is not present when the RSM is installed in a two-slot shelf, since the PEMs are not field replaceable units.

25.5 FRU Query Syntax

The format for querying the FRU of a particular location is:

```
cmmget -l <location> -t FRU -d <dataitem>
```

location is the component for which the FRU information is to be retrieved. *dataitem* specifies the field or fields of the FRU information to retrieve.

If you query the FRU of a particular location with the `cmmget` command, you can specify the location with no FRU ID appended to the location (for example, `blade5`) in order to retrieve the requested information (dataitem) for all the FRUs associated with the location specified in the command. On the other hand, if you specify a FRU ID (for example, `blade5:0`), the information retrieved is for the specified FRU only.

In either case, the appropriate FRU ID is prepended to the relevant information.

Here are some examples:

```
# cmmget -l chassis -t FRU -d all
FRU NAME: Chassis FRU
FRU TYPE: Chassis
CHASSIS TYPE: Rack Mount Chassis
PART #: MPCHC5089DC
SERIAL #: 1234567890
LOCATION: xxxxxxxxxxxxx
FRU NAME: Chassis FRU
FRU TYPE: Board
MANUFACTUREDATE: Mon Jan  1 00:00:00 1996
MANUFACTURER: Intel
DESCRIPTION: MPCHC5089
SERIAL #: ZZZZ12345678
PART #: C24328-102
FRU File ID: 103
FRU NAME: Chassis FRU
FRU TYPE: Product
MANUFACTURER: Intel
DESCRIPTION: MPCHC5089DC
```

```
PART #: MPCHC5089DC
REV. LEVEL:
SERIAL #: 1234567890
ASSET TAG:
FRU File ID:
# cmmget –l blade5 –t fru –d all
FRU NAME: 0:AMC Carrier
FRU TYPE: Board
DESCRIPTION: XXXXXXX
MANUFACTURER: Intel Corporation
PART #: 0000000000
SERIAL #: 000000000000
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
FRU NAME: 1:AMC Module
FRU TYPE: Board
DESCRIPTION: YYYYYYY
MANUFACTURER: Intel Corporation
PART #: 0000000001
SERIAL #: 000000000001
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
FRU NAME: 2:AMC Module
FRU TYPE: Board
DESCRIPTION: YYYYYYY
MANUFACTURER: Intel Corporation
PART #: 0000000001
SERIAL #: 000000000002
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
# cmmget –l blade5:0 –t fru –d all
FRU NAME: 0:AMC Carrier
FRU TYPE: Board
DESCRIPTION: XXXXXXX
MANUFACTURER: Intel Corporation
PART #: 0000000000
SERIAL #: 000000000000
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
# cmmget –l blade5:1 –t fru –d all
FRU NAME: 1:AMC Module
FRU TYPE: Board
DESCRIPTION: YYYYYYY
MANUFACTURER: Intel Corporation
PART #: 0000000001
SERIAL #: 000000000001
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
# cmmget –l blade5:2 –t fru –d all
FRU NAME: 2:AMC Module
FRU TYPE: Board
DESCRIPTION: YYYYYYY
MANUFACTURER: Intel Corporation
PART #: 0000000001
SERIAL #: 000000000002
MANUFACTUREDATE: Thu Dec  4 20:31:04 2003
# cmmget -l blade5 -t FRU -d boarddescription
0:AMC Carrier:XXXXXXX
1:AMC Module:YYYYYYY
2:AMC Module:YYYYYYY
# cmmget -l blade5:0 -t FRU -d boarddescription
0:AMC Carrier:XXXXXXX
# cmmget -l blade5:1 -t FRU -d boarddescription
1:AMC Module:YYYYYYY
# cmmget -l blade5:2 -t FRU -d boarddescription
2:AMC Module:YYYYYYY
```

Table 50, “Dataitems Used With FRU Target to Obtain FRU Information” lists the dataitems that can be used with the FRU target and the information they retrieve.

Table 50. Dataitems Used With FRU Target to Obtain FRU Information

Dataitem	Description
listdataitems	Displays a list of all FRU dataitems that can be queried for the FRU target and the given location.
all	Returns all FRU information for the location.
boardall	Lists all board area FRU information for the location.
boarddescription	Lists the name field in the FRU board area for the location.
boardmanufacturer	Lists the manufacturer field in the FRU board area for the location.
boardpartnumber	Lists the part number field in the FRU board area for the location.
boardserialnumber	Lists the serial number field in the FRU board area for the location.
boardfrufileid	Lists the FRU file ID field in the board area for the location.
boardmanufacturedatetime	Lists the manufacture date and time field in the FRU board area for the location.
productall	Lists all product area FRU information for the location.
productdescription	Lists the name field in the FRU product area for the location.
productmanufacturer	Lists the manufacturer field in the FRU product area for the location.
productpartnumber	Lists the part number field in the FRU product area for the location.
productserialnumber	Lists the serial number field in the FRU product area for the location.
productrevision	Lists the revision field in the FRU product area for the location.
productassettag	Lists the asset tag field in the FRU product area for the location
productfrufileid	Lists the FRU file ID field in the product area for the location.
chassisall	Lists all chassis area FRU information for the location. Must use the chassis location with this dataitem.
chassispartnumber	Lists the part number field in the FRU chassis area for the location. Must use the chassis location with this dataitem.
chassisserialnumber	Lists the serial number field in the FRU chassis area for the location. Must use the chassis location with this dataitem.
chassistype	List the type field in the FRU chassis area for the location. Must use the chassis location with this dataitem.

Note: Dataitems `productmodel` and `productmanufacturedatetime` are not supported as they do not map directly to FRU information storage fields.

25.6 Shelf Address

When listing all FRU information for the location “chassis”, there is a location field listed consisting of “xxxxx.”, which is not changeable. The correct chassis location information is kept in the Shelf Address record. Use the `location` dataitem on the chassis location to get and set the chassis location field. For example:

```
cmmget -l chassis -d location
```

Refer to “Alert Standard Format (ASF) Specification version 2.0” for more information.

Chapter 26

26.0 Command and Error Logging

The RSM logging service is based on the Linux `syslog` utility. The RSM relies on this service to provide user with logs on issued user commands, application errors, and debug information.

26.1 Log Levels and Facilities

The RSM logging service can be used to monitor RSM runtime behavior at five (5) different logging levels. These are:

- CRITICAL(4)
- ERROR(3)
- NOTICE(2)
- INFO(1)
- DEBUG(0)

Note: Level DEBUG is dedicated for debug mode logs that are visible only in debug firmware versions but filtered out in the release firmware version.

Rather than having a single logging level per system, the RSM supports separate logging levels per functionality. Each distinct functionality is identified by a facility name.

26.1.1 Environment Variables

The logging level is configurable. Environment variable CMM_LOG_LEVEL_DEFAULT controls the default RSM log level. If the environment variable is set, the log levels for all facilities are set to this value. Environment variable CMM_LOG_LEVEL_<facility> controls the log level for <facility>. If the environment variable is set, the log level for this facility is set to this value.

26.1.2 Log Level Control

Log levels can be controlled in run-time using a helper program, called `cmm_log_control`. This program allows the user to get and set all log levels for facilities in given RSM process(es). The program can be invoked as follows:

cmm_log_control [**-v**] [**-l**] [**-s** *level*] [**-n** *name*] {*facility* | ALL}

facility — Defines the unit of RSM functionality for which the log level can be set. Valid facility names can be listed by calling `cmm_log_control` without parameters. ALL stands for all facilities.

The options are:

-v — List facility names using verbose style.

-l — List log levels for the given *facility* in all RSM processes.

-s *level* — Set log level to *level* for the given *facility* in all RSM processes. Valid *level*:

- CRITICAL(4)
- ERROR(3)
- NOTICE(2)
- INFO(1)
- DEBUG(0)

-n *name*

Limits the scope of set/list commands to an RSM process executing program *name*. Valid *name* is:

- shm
- pm
- cmmget
- cmmset
- ntpd
- snmpd
- upgrade
- rmt_cli
- fru_update

26.2 Command Logging

All `cmmset` commands from all of the RSM interfaces (CLI, ShM API, and SNMP) are logged by the RSM in the command log file `/tmp/log/user.log` on RAM disk. When the command log reaches maximum size specified in `logrotate.conf`, the log file is compressed and archived using `gzip`, then stored in the `/var/log/cmm/cmm` directory on flash media. The format of the file name for the log files is `user.log.`*N*`.gz`, where *N* is the number of the log file archive. The maximum number of archives is configured in `logrotate.conf`. If the log file becomes full and there are already the maximum number of archives, the oldest archive is deleted to make room for the newest archive.

Caution:

- Archived files should *never* be decompressed on the RSM because the resulting prolonged flash file writing could disrupt normal RSM operation and behavior. Instead, the files should be transferred and decompressed on a different machine. Files can be decompressed by any application that supports the decompression of gzip (`*.gz`) file types.
- The `/var/log/cmm/cmm` directory should not be deleted or changed. The RSM requires that the directory exist to log errors.

26.3 Error Logging

Logging information for the RSM is dispatched between two log files: `error.log` and `debug.log`. The `error.log` and `debug.log` files are archived to maintain error logging in the event either log gets full and to prevent any loss of log data. This information is useful for technical support personnel.

26.3.1 error.log

RSM error logging information is logged in the file `/var/log/cmm/cmm/error.log` on flash media. When `error.log` reaches the maximum size specified in `logrotate.conf`, the log file is compressed and archived using `gzip`, then stored in the same directory. The format of the file name for the log files is `error.log.`*N*`.gz`, where *N* is the number of the log file archive. The maximum number of archives is configured in `logrotate.conf`. If the log file becomes full and there are already the maximum number of archives, the oldest archive is deleted to make room for the newest archive.

26.3.2 debug.log

Debug information for the RSM is logged in the file `/tmp/log/debug.log` on RAM disk. When `debug.log` reaches the maximum size specified in `logrotate.conf`, the log file is compressed and archived using `gzip`, then stored in the same directory. The format of the file name for the log files is `debug.log.`*N*`.gz`, where *N* is the number of the log file archive. The maximum number of archives is configured in `logrotate.conf`. If the log file becomes full and there are already the maximum number of archives, the oldest archive is deleted to make room for the newest archive.

26.4 Linux* logger

In addition to the above, the RSM logging service can be used to store user defined log entries using the Linux logger command. Linux command `logger(1)` makes entries in the system log. It provides a shell command interface to the `syslog(3)` system log module. The distribution package for version 8.x of the RSM firmware includes this command as part of the Linux distribution.

Note: This command is a standard utility in Linux and is *not* managed or controlled in any way by the RSM firmware.

The syntax of this command as supported in this release of the RSM firmware is:

logger [**-p** *pri*] [**-t** *tag*] [*message* ...]

The options are:

-p *pri*

Enter the message with the specified priority. The priority may be specified numerically or as a "facility.level" pair. For example, "-p local3.info" logs the message(s) as informational level in the *local3* facility. The default is "user.notice."

Valid facility names are: auth, authpriv (for security information of a sensitive nature), cron, daemon, ftp, mail, news, security (deprecated synonym for auth), syslog, user, uucp, and local0 to local7, inclusive.

Valid level names are: alert, crit, debug, emerg, notice, panic (deprecated synonym for emerg). For the priority order and intended purposes of these levels, refer to the Linux syslog(3) man page.

-t *tag*

Mark every line in the log with the specified *tag*

message

Write the message to log; if not specified, and the **-f** flag is not provided, standard input is logged.

The `logger` utility exits 0 on success, and >0 if an error occurs.

Note: The standard logger utility supports additional options. However, the options listed above are those that are supported in this release of the RSM firmware. Also, since logger runs as a user space process, logger is unable to log messages from the "kern" facility.

26.5 Configuring syslog

The behavior of the `syslog` utility is configured in the file `/etc/syslog-ng/syslog-ng.conf`. It is strongly recommended that the default configuration provided with the RSM firmware release in the `/etc/syslog-ng/syslog-ng.conf` file be maintained and that the log files be used as defined in that file.

For user specific purposes you can either use the existing log files or define your own log files. If you decide to use any of the existing log files, you should specify a unique tag with the "`-t`" option when logging to that file.

In order to maintain the performance of the RSM you should minimize logging to flash media (such as `/var/log/cmm`).

Note: Since syslog-ng is not a component that is managed by the RSM, the active RSM will not synchronize the `syslog-ng` configuration file to the standby RSM. The contents of this file also are not preserved during a firmware update. Modify this configuration file after completing the RSM firmware update to restore any changes you had made before the update. Whenever you modify the `syslog-ng.conf` file, you need to restart `syslog-ng` (see Section 26.5.2, "Restarting syslog-ng" on page 136).

26.5.1 Log Rotation and Archives

Log files can get rather large and cumbersome. Linux provides a command, `logrotate(8)`, for compressing and rotating log files so that current log information is not in the same file with older, less relevant data. Normally, `logrotate` runs automatically on a timed basis, but it can also be run manually. When run automatically, `logrotate` is executed as a `cron` job that runs (depending on the configuration) once a week, once a day, or once an hour.

When executed, `logrotate` takes the current version of the log file and append a ".1" to the end of the filename. Other previously rotated files are sequenced with the suffix ".2", ".3", and so on. The larger the number after a filename, the older the log is.

You configure the automatic behavior of `logrotate` by editing the `/etc/logrotate.conf` file. It is strongly recommended that you keep the default configuration provided with RSM distribution. However, you can define your own log rotation policy for your own log files.

Since `logrotate` is not a component managed by the RSM, the active RSM will not synchronize the `logrotate` configuration file to the standby RSM. Also, changes to the configuration file are not preserved during a firmware update. Modify the configuration file to restore any lost changes after the update. After modifying the contents of `logrotate.conf`, you need to restart syslog-ng or send it a SIGHUP signal (see Section 26.5.2, "Restarting syslog-ng" on page 136).

26.5.2 Restarting syslog-ng

If you decide to define your own logging policy by modifying the default `/etc/syslog-ng/syslog-ng.conf` file or the `/etc/logrotate.conf` file, you need to restart the syslog-ng service or send syslog-ng a SIGHUP signal after modifying either of those files.

Once you have modified the `syslog-ng.conf` file, you must either send syslog-ng a SIGHUP signal or restart syslog-ng to force syslog-ng to re-read the configuration file.

To send syslog-ng a SIGHUP signal, enter this command:

```
kill -HUP $(/sbin/pidof syslog-ng)
```

To stop and restart syslog-ng, do the following:

1. Kill syslog-ng with this command:

   ```
   kill $(/sbin/pidof syslog-ng)
   ```

2. Restart syslog-ng with this command:

   ```
   /etc/init.d/syslog-ng restart
   ```

The `logrotate.conf` file as distributed includes the command to send syslog-ng a SIGHUP signal after defining the rotation policy for `error.log` file. You can use these entries as an example of how to modify `logrotate.conf` to define a log rotation policy for other log files you use to capture output on an on-going basis.

26.5.3 Caveats and Limitations

If log files grow too large, the RSM may not be able to run properly or may hang. You are strongly advised to log only the *minimum number of messages* needed so that the log files do not grow too large, especially during the interval before `logrotate` runs to rotate and compress the log files.

Log files produced by `syslog` share flash storage in directory`/var/log/cmm` with SEL files and other diagnostic data such as the last reboot reason or crash log. In order to maintain the performance of the RSM, particularly if the log files are stored on flash media on the RSM board, the total size of log files (incl. archives) plus the size of SEL files (incl. archives) should not exceed 1920 kilobytes.

As stated previously, the recommended action is to keep the default configurations and files as they are defined in the RSM firmware distribution package. Nonetheless, if you decide to modify those configuration files or use different files for logging, you should avoid creating your log files in the /etc file system, or anywhere under /usr/share/cmm/scripts. The preferred location is /tmp/log.

If you write the log messages to a file on an NFS-mounted filesystem, be aware that the filesystem will not be unmounted automatically after the current messages have been written. This is because the syslog-ng daemon on Linux does not perform an automatic umount after completing the write operation. You must manually unmount the filesystem yourself.

The guideline to avoid creating log files anywhere under /usr/share/cmm/scripts is especially important since all files in this directory are synched from the active RSM to the standby RSM to maintain consistent information on both RSMs. Data synching should not occur more often than necessary and the size of the files to be synched should also be small. The presence of the log files in this directory will add to the load of the synchronization process.

Chapter 27

27.0 Diagnostics

27.1 U-Boot Diagnostic Tests

The implementation of U-Boot on the RSM supports two kinds of diagnostic tests: POST diagnostics and Manufacturing diagnostics. POST diagnostics are tests that are run during the board's initialization to verify whether or not the board is healthy enough to boot to Linux. Manufacturing diagnostics are typically more invasive or time-consuming tests that can be used by Manufacturing to test the robustness of a board or to debug issues.

U-Boot generates System Firmware progress events to the shelf manager to indicate boot-up information. See Table 74 on page 207 and the *A6K-RSM-J Shelf Manager Hardware Reference* for information about the events generated by the Sys FW Progress sensor.

This section describes the different diagnostic options that are available on the RSM's U-Boot implementation.

27.1.1 BOARD_INIT_RAM_TEST

When the power comes out of reset, U-Boot initially runs out of the LMP's local L2 SRAM/cache. After it has configured the external DDR memory, U-Boot transfers itself to the DDR memory so that it has more operational resources. Before U-Boot transfers itself to DDR memory, it performs tests on the memory to make sure it is operating properly. If the memory is not functioning, U-Boot may hang or events will be generated.

The tests that run before U-Boot copies itself to RAM are defined in the U-Boot environment variable `BOARD_INIT_RAM_TEST`. By default, this variable is set up to run the POST test `LMPpostmtest` on a small range of memory. The variable can be changed if more in-depth testing is required.

27.1.2 POST Diagnostics

POST diagnostics are tests that run as the last step of the U-Boot initialization process. These tests are designed to run quickly. POST diagnostics are any U-Boot test command with the value "post" in the name. Each POST diagnostic test verifies a minimal amount of functionality in a given area.

The environment variable `postdiagscold` defines the set of POST tests to execute. The contents of this variable can be modified, if desired.

By default, U-Boot verifies that I2C devices are responding, Ethernet connections are physically working, and MAC IDs are specified.

The POST tests are described in detail in the following sections.

27.1.2.1 LMPpostmtest

This test verifies the memory caches and SRAM for the LMP and the LMP processor core complex. This test validates 8 KB of memory on either side of each 1 MB boundary in the specified memory range. It writes different patterns on each side of the boundary and then reads the values.

This test is based on the LMPmtest function.

Syntax:

```
LMPpostmtest <start-addr> <stop-addr>
```

Command options:

```
<start-addr> Specifies the starting address to test, from 0x0 to
  0x3f00_0000

<stop-addr> Specifies the ending address to test, from 0x01 to
  0x3f00_0000
```

27.1.2.2 LMPposti2ctest

This test scans for all expected devices on I2C bus 1 and verifies that all expected devices respond.

Syntax:

```
LMPposti2ctest
```

27.1.2.3 LMPpostmactest

This test verifies that MAC addresses in the MAC EEPROM have been configured to a non-0xFF value.

This test is based on the LMPmactest function.

Syntax:

```
LMPpostmactest
```

27.1.2.4 LMPpostethtest

This test verifies that the LMP can access each of the board's Ethernet ports via U-Boot. The test does not verify whether traffic can be passed through the devices.

Syntax:

```
LMPpostethtest
```

27.1.3 Manufacturing Diagnostics

Manufacturing diagnostics are similar to POST diagnostics, but manufacturing diagnostics have the potential to be more invasive and time consuming.

The manufacturing tests are described in detail in the following sections.

27.1.3.1 LMPintmemtest

This test verifies memory caches and SRAM for the LMP and the LMP processor core complex.

Syntax:

```
LMPintmemtest <pattern-type> [<iteration-count> <stop-on-error>]
```

Command options:

`<pattern-type>` Specifies the type of test to perform. The possible values are:

0	Performs all memory tests
1	Writes simple pattern to memory
2	Tests addressability by walking 1s and 0s across the address bus
3	Tests the data bus by walking 1s and 0s across the data bus

27.1.3.2 LMPipmctest

This test verifies that the LMP access to the IPMC UART port is functional by sending and receiving the **`Get Device ID`** command.

Syntax:

```
LMPipmctest [<iteration-count> <stop-on-error>]
```

27.1.3.3 LMPnandtest

This test verifies that the NAND Flash Controller (NAND FPGA) and Radisys U-Boot NAND driver are correctly identifying and correcting ECC errors. The test injects errors into flash with known data by temporarily disabling ECC in the NAND FPGA. The RSM supports 4-bit ECC protection, which means that injecting five errors causes the block under test to be marked as bad. Use this command with discretion as it has the potential to permanently wear out a block of NAND Flash.

Syntax:

```
LMPnandtest <pattern-type> <nand offset> [<iteration-count> <stop-on-
  error>]
```

Command options:

`<pattern-type>` Specifies the type of test to perform. The possible values are:

1	Injects one error into each 512-byte block of data in a page
2	Injects two errors into each 512-byte block of data in a page
3	Injects three errors into each 512-byte block of data in a page
4	Injects four errors into each 512-byte block of data in a page
5	Injects five errors into each 512-byte block of data in a page

`<nand offset>` Offset in NAND from which to perform the test

27.1.3.4 LMPmtest

This test has the same interface and description as `LMPpostmtest`.

27.1.3.5 LMPmactest

This test is has the same interface and description as `LMPpostmactest`.

27.1.3.6 LMPethtest

This test is has the same interface and description as `LMPpostethtest`.

27.2 Run-Time Diagnostics

The RSM supports non-destructive diagnostics in run-time. Those tests check the operational state of selected devices while the RSM is in service.

27.2.1 Flash Diagnostics

Flash test scans the flash partitions holding images. For each partition, the test makes a raw read and calculates a CRC32 checksum on the image stored in the partition. The recalculated image checksum is then compared to the one stored on the flash in the image trailer. If at least one checksum is not correct the test fails, otherwise it ends with success.

To run flash diagnostics, execute the following CLI command:

```
cmmset -d TestFlash -v start
```

27.2.2 Ethernet Diagnostics

The Ethernet test verifies Ethernet connectivity. ICMP ping is performed using the OS `ping` utility, specifying the destination IP address supplied in the request parameter.

To run the Ethernet test, execute the following CLI command:

```
cmmset -d TestEth -v <ipaddress>
```

27.3 Reboot Reason Discovery

The RSM discovers and persists the reason of the last reboot on its own. You can learn the reason of the last RSM reboot by querying the “Reboot Reason” sensor. For a detailed definition of sensor states, refer to Appendix D, “OEM Sensor Events”.

The reason for the last reboot may be software operations which are controlled by the system, such as system upgrade or OS shutdown. Those reasons are stored in a file system in the `/var/log/cmm/cmm/last_reboot_reason` file.

The `/var/log/cmm/cmm/last_reboot_reason` is subject to log rotation through `logrotate`. Configuration is stored in `/etc/cmm/logrotate_crashlog.conf`.

27.4 RSM Crash Logging

By default, the OS is configured to not produce core files on a process crash. This is because the persistent storage space is scarce. RSM processes generate small crash logs when they terminate unexpectedly due to a malfunction. The system operator can collect crash logs and send them to Radisys support for analysis. The operator can also send a malfunctioning (hung) RSM process a SIGSEGV signal, causing it to produce the crash log and terminate. The same action can be performed by Radisys support working on a customer's site to pinpoint the problem.

In order to obtain some debugging information, every RSM process links with a library, which defines the handler for the following OS signals:

- SIGSEGV
- SIGBUS
- SIGILL
- SIGABRT

To activate RSM crash logging, DUMPSIZE variable in `/etc/cmm/core.config` must be set to 0 (this is the default value).

When an RSM process is terminated by the OS due to an illegal operation, the crash handlers dump as much information as possible about the currently executing (and faulting) thread.

On its startup, the library allocates sufficient memory to store up to 50 stack frame pointers (of type void*) and installs handlers for SIGSEGV, SIGBUS, SIGABRT and SIGILL signals.

When invoked, the handler takes the following steps:

1. Opens a binary file, named after `<program_name>-<PID>` in `/var/log/cmm/cmm/crash`
2. Write a timestamp and output of `uname -a` to the above file
3. Dump contents of all CPU registers to the above file
4. Dump the list of stack frame pointers to the above file
5. Receive the faulting function frame pointer
6. Close the file
7. Invoke the default signal handler, which terminates the process

27.5 Core Dump

Core dumps are disabled by default because of lack of storage. A system administrator must mount an external NFS storage for core files and then the system operator can enable core dumps as described below. An operator can also force any OS process to terminate and produce a core dump by sending it a SIGSEGV signal. Core dumps are then analyzed by Radisys.

The Linux kernel allows dumping core files to specified locations and naming them in a unique way. `/etc/cmm/core.config` - can be modified by the user and contains the following variables:

DUMPFORMAT - format of the core file name, as described in the Linux kernel documentation.

DUMPLOCATION - directory location of the core file. The location should be a mounted, writable NFS volume or other permanent storage other than the RSM flash because the available flash space is limited. The user is responsible for mounting the volume.

DUMPSIZE - maximum size of the core file, set to a value greater than 0 by default. To disable core dumps and active crash dumps, set this parameter to 0.

Changes in `/etc/cmm/core.config` become effective after the next reboot.

27.6 Kernel Crash Logging

Kernel crash logging is a debugging capability that appends the contents of the kernel system log ring buffer to a reserved block of flash memory. It provides a way of capturing debug and trace data without using serial port consoles or custom kernel drivers.

27.6.1 Kinds of Data Logged

This logging feature appends the kernel log buffer to the flash memory when certain events occur, such as a kernel panic, oops messages, and software watchdog timer time-outs. In addition to the contents of the kernel log buffer, this feature appends the processor register set information.

27.6.2 Accessing Logged Data

If the RSM reboots due to a kernel panic, the kernel saves its log ring on flash partition `/dev/mtd9`. On system startup, the OS startup script `S03crashlog` checks if the crash log exists. If it exists, it copies its contents to the `/var/log/cmm/cmm/crash/kernel_panic.log` file. After that, the reserved flash block is erased.

27.6.3 Kernel Crash Log Rotation

The `kernel_panic.log` is subject to log rotation through `logrotate`. The configuration is stored in `/etc/cmm/logrotate_crashlog.conf`.

27.6.4 Sample Log File

```
<0>Kernel panic: /dev/sys/panic: panic test
<4> <0>strat dump from panic.c line 100
<3>kstat at xtime.tv_sec = 1124190273
<3> idle = 0 <3> per_cpu_user = 0 <3> per_cpu_nice = 0 <3> per_cpu_system =
100
<3> context_switch = 0
<3> irqs[0] = 0
<3> irqs[1] = 0
<3> irqs[2] = 0
<3> irqs[3] = 0
<3> irqs[4] = 0
<3> irqs[5] = 0
<3> irqs[6] = 0
<3> irqs[7] = 0
<3> irqs[8] = 0
<3> irqs[9] = 100
<3> irqs[10] = 0
<3> irqs[11] = 0
<3> irqs[12] = 0
<3> irqs[13] = 0
<3> irqs[14] = 0
<3> irqs[15] = 0
<3> irqs[16] = 0
<3> irqs[17] = 0
<3> irqs[18] = 0
<3> irqs[19] = 0
<3> irqs[20] = 0
<3> irqs[21] = 0
<3> irqs[22] = 0
<3> irqs[23] = 0
<3> irqs[24] = 0
<3> irqs[25] = 0
<3> irqs[26] = 0
<3> irqs[27] = 0
<3> irqs[28] = 0
<3> irqs[29] = 0
```

```
<3> irqs[30] = 0
<3> irqs[31] = 0
<3> irqs[32] = 0
<3> irqs[33] = 0
<3> irqs[34] = 0
<3> irqs[35] = 0
<3> irqs[36] = 0
<3> irqs[37] = 0
<3> irqs[38] = 0
<3> irqs[39] = 0
<3> irqs[40] = 0
<3> irqs[41] = 0
<3> irqs[42] = 0
<3> irqs[43] = 0
<3> irqs[44] = 0
<3> irqs[45] = 0
<3> irqs[46] = 0
<3> irqs[47] = 0
<3> irqs[48] = 0
<3> irqs[49] = 0
<3> irqs[50] = 0
<3> irqs[51] = 0
<3> irqs[52] = 0
<3> irqs[53] = 0
<3> irqs[54] = 0
<3> irqs[55] = 0
<3> irqs[56] = 0
<3> irqs[57] = 0
<3> irqs[58] = 0
<3> irqs[59] = 0
<3> irqs[60] = 0
<3> irqs[61] = 0
<3> irqs[62] = 0
<3> irqs[63] = 0
<3> irqs[64] = 0
<3> irqs[65] = 0
<3> irqs[66] = 0
<3> irqs[67] = 0
<3> irqs[68] = 0
<3> irqs[69] = 0
<3> irqs[70] = 0
<3> irqs[71] = 0
<3> irqs[72] = 0
<3> irqs[73] = 0
<3> irqs[74] = 0
<3> irqs[75] = 0
<3> irqs[76] = 0
<3> irqs[77] = 0
<3> irqs[78] = 0
<3> irqs[79] = 0
<3>forcing hardware WDT to go off now
<6>SysRq : Show Regs
<4>pc : [<c0022150>] lr : [<00000000>] Not tainted
<4>sp : c7b7bf44 ip : 00000000 fp : c7b7bf50
<4>r10: 4015082c r9 : c7b7a000 r8 : 40018000
<4>r7 : 00000009 r6 : c012ef88 r5 : c012efa8 r4 : c0193fec
<4>r3 : 00000000 r2 : c018689c r1 : 00000000 r0 : c0186890
<4>Flags: nZCv IRQs on FIQs on Mode SVC_32 Segment user
<4>Control: 197F Table: A7930000 DAC: 00000015
<6>SysRq : Emergency Sync
```

27.7 cmmdump Utility

The `cmmdump` utility is a script that captures important system information from the RSM system that can be helpful to support personnel in isolating the cause of a problem.

This utility is executed from a shell prompt on the RSM. The output is sent to the standard output and any errors are sent to the standard error. Both can be redirected to a file to log the data and any errors, as follows:

`cmmdump &> filename`

Because the resulting file can be quite large, you should capture the file in one of the following ways:

- Mount a remote storage device on the RSM file system using NFS (Network File System) and store the output file on that device.
- Capture the output that is sent to the standard output of your login session using the Capture Text or similar functionality in your client console program.
- Redirect the output to a file on the RAM disk in `/tmp`.

Note: If you redirect the output to the RAM disk, the file should then be transferred from the RSM to another storage device as soon as possible. This is important to avoid filling up the RAM disk since the RSM firmware and other components use the RAM disk for storage. In any case, you must transfer the file before the RSM reboots, since a reboot clears the RAM disk.

27.8 Operating System Flash Corruption Detection & Recovery

The operating system is responsible for the flash content integrity at runtime. Flash monitoring under the operating system environment can be divided into two parts: Monitoring static images and monitoring dynamic images.

Static images refer to the U-Boot image, rootfs image, and Linux image in flash memory. These images should not change throughout the lifetime of the RSM unless they are purposely updated or corrupted. The checksum for these files is written into flash memory when the images are uploaded.

Dynamic image refers to the operating system Flash File System (JFFS2). This image dynamically changes during execution of the operating system.

27.8.1 Monitoring Static Images

Flash test is run periodically (i.e. every 24 h) while the RSM firmware is running. The static test reads each static image, calculates the image checksum, and compares the calculated checksum with the checksum stored in the image header. If the checksums do not match, the error is logged to the system log.

27.8.2 Monitoring Dynamic Images

For monitoring the dynamic images, the RSM leverages the corruption detection ability of the JFFS(2) flash file system. At operating system start-up the RSM executes an initialization script to mount the JFFS(2) flash partitions `/etc/cmm` and `/usr/share/cmm` and `/var/log/cmm`. If corruption of the flash memory is detected, an event is logged to the system log.

During normal operating system operation, flash corruption during file access can also be detected by either the JFFS(2) or the flash memory driver. If corruption of the flash memory is detected, an event is logged to the system log.

Chapter 28

28.0 Statistics

Apart from OEM sensors, the RSM provide statistics readable by the System Management interfaces (SNMP, CLI, ShM API) for various data relevant to its health and performance. The following types of statistics are provided:

- **Counters** - incremented every time some event takes place (e.g., on the reception of the incoming frame)
- **Gauges** - numerical values fluctuating over time (e.g., system load)
- **Second order statistics** - computed values derived from the first order counters or gauges. The general rule is that there is a very limited amount of second order statistics, relevant to the overall system health. More complicated and not critical second order statistics should be computed by the client.

Some of the counters and gauges support configurable thresholds (either upper, lower, or both). When the threshold is reached, an event is generated to the system log.

28.1 Querying Statistics Values

Statistics are organized into groups per functional area. All OS-related statistics are organized into one group. To get the list of supported groups, execute the CLI command:

```
cmmget -t stats -d list
```

To get the names of all statistics in a particular group, execute the command:

```
cmmget -t stats:<group> -d list
```

where <group> is one of a valid group of names listed as an output from the first command.

To get the value and thresholds of a selected statistic, execute the command:

```
cmmget -t stats:<group>:<name> -d show
```

where <group> is one of a valid group of names, and <name> is a valid statistics name within the indicated <group>.

For example, query IPMI generic statistic "ResponseQueued" with the following command:

```
cmmget -t stats:IpmiGeneric:ResponseEnqueued -d show
```

To reset the reading of a selected statistic, execute the command:

```
cmmset -t stats:<group>:<name> -d reset -v 1
```

where <group> and <name> are defined as above.

If a statistic supports thresholds, they can be changed. To set a threshold on a selected statistic, execute the command:

```
cmmset -t stats:<group>:<name> -d threshold -v <type>:<value>
```

where <group> and <name> are defined as above, <type> is the threshold type (upper, lower), and <value> is the threshold value.

Note: Collected statistics data is not replicated between an active and standby RSM.

28.2 OS Statistics

The OS statistics group supports the following statistics:

- **Load_Average_1** - average system load in the last minute. Obtained by reading /proc/loadavg. Multiplied by 100.
- **Load_Average_5** - average system load in the last 5 minutes. Obtained by reading /proc/loadavg. Multiplied by 100.
- **Load Average_15** - average system load in the last 15 minutes. Obtained by reading /proc/loadavg. Multiplied by 100.
- **FS_<device>** - file system usage. Multiple counters of this type exist, one for each mounted JFFS file system. The <device> is the name of the flash partition containing the file system.
- **Mem_Total** - total amount of memory.
- **Mem_Free** - free memory.

For example, query the OS statistic "Load_Average_1" with the following command:

```
cmmget -t stats:OS:Load_Average_1 -d show
```

Note: The OS statistics do not allow setting thresholds.

Appendix E, “Statistics” on page 286 lists all supported statistics.

Chapter 29

29.0 Time Synchronization

Time Synchronization provides the following functionality:

- Synchronization of the local clock to external time servers
- Synchronization of the standby RSM clock to the active RSM clock
- Optionally can provide clock synchronization to other blades in the chassis

To provide this functionality, the Time Synchronization module implements the Network Time Protocol daemon (ntpd), which communicates to other time servers and clients over the network connection.

Clock synchronization between active and standby RSMs is achieved running NTP over IPMB using a proprietary encapsulation format.

Time Synchronization uses NTP version 3 [*RFC1305*].

To check the operational status of Time Synchronization, execute the command:

```
cmmget -t TimeSync -d Status
```

To change the operational status of Time Synchronization, execute the command:

```
cmmset -t TimeSync -d Status <status>
```

where *status* is Enable or Disable. Disabling Time Synchronization has no impact on clock synchronization between Active and Standby.

29.1 Default Configuration

Time Synchronization is turned on by default. In the default configuration, only the time synchronization of the active RSM clock with the standby RSM clock is operable. The list of external NTP servers is empty. The list of broadcast addresses is empty. The list of local listen addresses is empty.

29.2 Configuring NTP Client

The NTP client synchronizes its clock to an external NTP timeserver. The NTP client may be configured to use multiple NTP timeservers. It is possible to set a preference for a specific NTP timeserver as the most accurate time source. There are several publicly accessible NTP timeservers on the Internet. See http://ntp.isc.org/bin/view/Servers/WebHome for more details. The address of the external NTP timeserver is configured using this CLI command:

```
cmmset –t TimeSyncServer:<index> -d Add –v <address>:<port> [,<preferred>
[,<NTP version>[, <minPoll>[, <maxPoll>]]]]
```

Table 51. Add NTP server address - CLI command parameters

name	description
Index	(mandatory) server index: 0-9
Address	(mandatory) server IP address, e.g. 128.101.20.1
Port	(mandatory) server TCP port number: 0-65535
preferred	(optional) if set to true this peer is a preferred clock source. Preferred server responses are discarded only if they vary dramatically from other time sources. Otherwise, the preferred server is used for synchronization without consideration of the other time sources. Mark the server as the preferred one if it is known to be extremely accurate. Allowed values: 0 – not preferred clock source (default) 1 – preferred clock source
NTP version	(optional) NTP version used in communication with this server. Allowed values: • 2 • 3 (default)
minPoll	(optional) Minimum polling interval for this server. Allowed values: 16, 32, 64 (default), 128, 256, 512, 1024.
maxPoll	(optional) Maximum polling interval for this server. Allowed values: 16, 32, 64, 128, 256, 512, 1024 (default).

The configured address of the existing NTP timeserver can be removed using the CLI command:

```
cmmset –t TimeSyncServer:<index> -d delete –v 1
```

Table 52. Delete NTP server address - CLI command parameters

name	description
index	(mandatory) server index: 0-9

A specific NTP timeserver entry can be displayed using the CLI command:

```
cmmget –t TimeSyncServer:<index> -d Show
```

Table 53. Show NTP server address entry - CLI command parameters

name	description
index	(mandatory) server index: 0-9

Below is example output for this command:

```
> cmmset –l cmm –t TimeSyncServer:1 –d Show
 Server address: 128.101.20.1:1000
 NTP version: 3
 Min poll interval: 64
 Max poll interval: 1024
 Preferred server: True
```

29.3 Configuring NTP Server

The RSM may act as an NTP timeserver, providing its time as a reference to other NTP nodes in the network. For example, SBC blades in the chassis may use an NTP server running on an RSM as the source of the reference clock. The NTP server listens to the incoming NTP time synchronization requests on local listen addresses. The NTP server local listen address can be configured using the CLI command:

```
cmmset –t TimeSyncListen:<index> -d Add –v <address>:<port>
```

Table 54. Add NTP listen address - CLI command parameters

name	description
index	(mandatory) Time Synchronization Listen address index: 0-4
address	(mandatory) Local IP address, e.g. 128.101.20.1
port	(mandatory) TCP port number: 0-65535

The configured NTP server local listen address can be deleted using CLI command:

```
cmmset –t TimeSyncListen:<index> -d Delete –v 1
```

Table 55. Delete NTP listen address - CLI command parameters

name	description
index	(mandatory) Time Synchronization Listen address index: 0-4

A specific NTP local listen address entry can be displayed using the CLI command:

```
cmmget –t TimeSyncListen:<index> -d Show
```

Table 56. Show NTP client address entry - CLI command parameters

name	description
index	(mandatory) Time Synchronization Listen address index: 0-4

For example:

```
> cmmset –t TimeSyncListen:1 –d Show

  128.101.20.1:1000
```

29.4 Configuring NTP Server in Broadcast Mode

In broadcast mode, an NTP server periodically broadcasts its time setting over the network using NTP packets addressed to a configured broadcast IP address. Any NTP client that can receive these broadcast packets may use them to synchronize its time. The broadcast address for an NTP server can be configured using the CLI command:

```
cmmset –t TimeSyncBcst:<index> -d Add –v <address>:<port>,<interval>
```

Table 57. Add NTP broadcast address - CLI command parameters

name	description
index	(mandatory) Time Synchronization Broadcast address index: 0-4
address	(mandatory) Broadcast IP address
port	(mandatory) TCP port number: 0-65535
interval	(mandatory) Specifies the interval for sending out broadcast NTP messages to the specified address. The interval is specified in seconds. Allowed values are: 16, 32, 64 (default), 128, 256, 512, 1024.

The configured broadcast address can be deleted using the CLI command:

```
cmmset –t TimeSyncBcst:<index> -d Delete –v 1
```

Table 58. Delete NTP broadcast address - CLI command parameters

name	description
index	(mandatory) Time Synchronization Broadcast address index: 0-4

The configuration of a specific NTP server broadcast address entry can be displayed using the CLI command:

```
cmmget –t TimeSyncBcst:<index> -d Show
```

Table 59. Show NTP broadcast address entry - CLI command parameters

name	description
index	(mandatory) Time Synchronization Broadcast address index: 0-4

For example:

```
> cmmget –t TimeSyncBcst:1 –d Show

128.101.255.255:1000 interval: 128
```

29.5 Time Synchronization Sensor

The “Time Synchronization“ Sensor provides means to receive information about the state of the local clock, i.e. whether it stays properly synchronized to the specified clock server. The “Time Synchronization” Sensor layout is defined in Appendix D, “OEM Sensor Events”.

29.6 RTC Synchronization

NTP controls the system clock by updating its setting according to the information received from the network. Whenever the system clock setting is changed by the NTP, the RTC should be updated accordingly. An RTC udate also happens after each reboot and use of the `setdate` command. It is up to the Linux* kernel to synchronize the system clock setting with the RTC. Every 11 minutes inside of the timer interrupt Linux triggers the RTC synchronization procedure.

29.7 Configuration File

Configuration of Time Synchronization module is stored in configuration file `/etc/cmm/timesync.conf`. By default, the configuration file is empty.

30.0 Setting Up the RSM

30.1 Connecting to the RSM

The RSM provides two physical Ethernet connections on its front panel and two Ethernet connections through the rear backplane connector. The front panel connections are made via an RJ-45 connector.

Note: If you are logging in for the first time to set up or obtain the RSM's IP addresses, you must use the serial port console interface to perform configuration.

Any of these interfaces can be used to log into the RSM. Use the `telnet` application to log into the RSM over an Ethernet connection or use a terminal application or serial console over the RS-232 interface. See the "A6K-RSM-J Hardware Reference" for the electrical pinouts of the above interfaces.

30.2 Initial Setup

Logging in for the first time must be done through the serial port console to properly configure the Ethernet settings and IP addresses for the network.

Connect an RS-232 serial cable with an RJ-45 connector to the serial console port on the front of the RSM. Set your terminal application settings as follows:

- Baud rate – 115200
- Data Bits – 8
- Parity – None
- Stop Bits – 1
- Flow Control – Xon/Xoff or none

Connect using your terminal emulation application.

The username when logging in to the RSM is `root`. The default password is `cmmrootpass`.

At the login prompt, enter the username: `root`

When prompted for the password, enter: `cmmrootpass`

The root password can be changed using CLI command. For details refer to Chapter 13.0, "Security". The root password can be set back to the default `cmmrootpass`. For information on resetting the RSM password back to the default, refer to Chapter 13.0, "Security".

30.2.1 Setting IP Address Properties

It is extremely important to correctly configure the connection of the RSMs to the network in order for the RSMs to function properly and manage the components in the chassis.

The OS network stack of the RSM is initialized as part of the OS load before RSM software stack initialization. At this first network stack initialization, the network data from the Chassis Data Module is not available. This initial start of the OS network stack uses the factory default configuration in the `/etc/sysconfig/network-scripts/ifcfg-ethx` file, where ethx can be eth0, eth1, eth2, or eth3. Once the RSM is up, the network settings can be changed using the system management interface method in Chapter 31.0, "IP Network Configuration".

Caution: The manual method of setting network configuration data (using the `vi` editor) is not supported. You should avoid doing manual modifications as there is no guarantee that the changes will be propagated into the Shelf FRU and OS network stack.

30.2.2 Setting a Hostname

The hostname of the RSM is a logical name that is used to identify a particular RSM. This name is shown at login time just to the left of the login prompt on the serial port interface when configured (for example, “MYHOST login:”) The hostname is advertised to any DNS servers on a network.

The hostname is set in the `/etc/cmm/hostname` file. The hostname is persistent and takes effect on the next boot.

The hostname is changed using this command:

```
hostname some_host
```

Note: The changed hostname is not persistent across reboots if the `hostname` command is used.

The current hostname is displayed using this command:

```
hostname
```

30.2.3 Mounting NFS

The user can mount NFS volumes. To minimize the system CPU load caused by NFS processing and to assure stable operation of RSM software, NFS volumes should be mounted with maximum available read/write buffer size.

30.2.4 Setting Time for Auto-logout

For security purposes, the RSM automatically logs the user out of the current console session after a period of inactivity. The length of this period can be changed by editing `/etc/profile` and changing the time-out (TMOUT) value. The time-out value is set in seconds, and 900 seconds (15 minutes) is the default. A setting of TMOUT=0 disables the automatic logout.

Note: As with all shell variables, this variable can also be modified from the shell prompt.

30.2.5 Setting Date and Time

To view the current date and time execute the `date` Linux command. To set the date and time execute the `date` Linux command as follows:

`date -s "mm/dd/yyyy [`*timezone*`] hh:mm:ss"`

The timezone can be included in the `date` string. The RSM determines the offset to the local timezone maintained in file `/etc/cmm/TZ` and automatically updates the time.

Note: The date and time must be set to any valid date and time after 00:00:00 UTC, January 1, 1970.

After setting the date and time, execute the following command to synchronize the date and time with the real time clock (RTC):

```
hwclock --systohc
```

The following example sets the date and time to Mar 11 20:12:00 UTC 2006:

```
date -s “03/11/2006 UTC 20:12:00”
```

Instead of "`date -s`" the setdate command from previous firmware versions can also be used with the same parameters as in "`date -s`".

Use these commands only on the active RSM.

Continuous time and date synchronization is handled using the NTP (RFC-1305) client-server synchronization model. Refer to *Time and Date Synchronization* on page 54 for more details on time and date synchronization.

Refer to *Time Synchronization* on page 148 for more details on RSM time management.

30.2.6 Establishing an Interactive Session

To establish an interactive session with the RSM firmware, connect the console or telnet application to the IP address of the eth0, eth1, eth2, eth3, or eth1:1 interface on the RSM. To connect to the active RSM use the eth1:1 IP address. To get the IP address, use methods described in *IP Network Configuration* on page 156.

30.2.7 Connect through SSH

The RSM firmware distribution package includes several components of the SSH (secure shell) protocol. The SSH components supplied provide support for secure remote login, secure file transfer and file copying. SSH can automatically encrypt, authenticate, and compress transmitted data.

The supplied components support version 2 of the SSH protocol.

30.2.7.1 Components

The components provided can log into another computer over a network, execute commands on a remote machine, and move files from one machine to another. They provide strong authentication and secure communications over insecure channels. They are secure replacements for the rlogin, rsh, and rcp executables.

The components supplied are:

- ssh—Client login program
- sshd—Daemon (server) that accepts login requests from ssh
- sftp—Secure FTP program
- scp—Secure file copy program
- ssh_config—Configuration file for ssh
- sftp-server—Server subsystem that responds to requests from sftp (located in `/usr/sbin`)
- ssh-keygen—Key generation tool
- ssh-rand-helper—Random number gatherer (located in `/usr/sbin`)
- ssh-prng_cmds—Contains paths to a number of files that ssh-keygen may need to use since the operating system provided with the RSM firmware package does not have a built-in entropy pool (like `/dev/random`). This file also contains commands to gather entropy for the OpenSSH pseudo-random number generator.

All of the components (except ssh-rand-helper) are part of OpenSSH. You can visit their web site at:

`http://www.openssh.com`

30.2.7.2 Initialization

When version 8.x of the RSM firmware is first installed, part of the initialization of SSH includes the initialization of the RSA and DSA host keys to be used for encryption. These keys are stored in the `/etc/ssh` directory.

During this initialization process, you see messages such as the following:

```
Generating SSH1 RSA host key:OK

Generating SSH2 RSA host key:OK

Generating SSH2 DSA host key:OK

Starting SSHD Service:OK
```

Once the initialization is complete, use the SSH client to open the IP address of the eth0, eth1, eth2, eth3, or eth1:1 interface on the RSM that will be used to establish an SSH session.

30.2.7.3 Further Information

To learn more about the SSH components supplied, refer to the online manual pages at:

http://www.openssh.com/manual.html

The manual page for ssh-rand-helper can be found at this site:

http://downloads.openwrt.org/people/nico/man/man8/ssh-rand-helper.8.html

30.2.8 Rebooting the RSM

To reboot the RSM, execute the `reboot` command on the RSM that is to be rebooted.

If the reboot command is executed on the active RSM in a redundant configuration, a failover to the standby RSM occurs. If the reboot command is issued on an RSM in a single RSM configuration, chassis management is unavailable during the reboot process. Telnet and SSH sessions will have to be reestablished with the RSM after it is rebooted.

Caution: Do not use the `init 0` or `init 6` commands to reboot the RSM.

Chapter 31

31.0 IP Network Configuration

31.1 Introduction

The RSM requires several pieces of information in order to utilize its available network interfaces. In a redundant (dual RSM) configuration this information includes:

- IP address of the active RSM
- netmask for the active RSM
- default gateway for the active RSM
- eth0, eth1, eth2, and eth3 IP addresses of both RSMs
- eth0, eth1, eth2, and eth3 netmask for both RSMs
- eth0, eth1, eth2, and eth3 gateway for both RSMs
- eth0, eth1, eth2, and eth3 boot protocol for both RSMs

Network information is stored in the following locations:

- Shelf FRU records stored on Chassis Data Module(s). This is the primary location for this data.
- The configuration files: `/etc/sysconfig/network-scripts/ifcfg-ethx` and `/etc/cmm/networks.conf`. This is the backup location for network data. The RSM uses the backup storage in case the information in the Shelf FRU cannot be retrieved.
- OS network stack

31.2 Shelf Manager IP Connection Record

The Shelf Manager IP Connection Record defined by the PICMG* 3.0 Specification is used to store the network configuration information for the active RSM (items 1 to 3 on the list above). These records are stored in the Shelf FRU MRA (MultiRecord Area), as defined in the Platform Management FRU Information Storage Definition v1.0 R 1.1.

There are two different formats defined for the Shelf Manager IP Connection Record: a base format (type 0x00) defined in the base specification (PICMG 3.0 R 1.0), and a newer format (type 0x01) defined in the Engineering Change Notice, ECN 001. The base format can store only the IP address information, whereas the newer format defined in ECN 001 can store the netmask and gateway information in addition to the IP address. The RSM supports both of these formats.

The Shelf Manager IP Connection Records must first be defined in the MRA of the Shelf FRU before network configuration information can be stored into and retrieved from the Shelf FRU. To define those records, either ensure that the `fru_update` utility runs as part of the RSM firmware update process or run the `fru_update` utility separately. For more information about the `fru_update` utility, see Chapter 34.0, “FRU Update Utility” on page 176.

Note: If the Shelf Manager IP Connection Record in the Shelf FRU uses the base format (type 0x00), only the IP address can be stored in the Shelf FRU. If this is the case, the `cmmget` command will return only the IP address, and the `cmmset` command will accept only the IP address in the value string argument to the `-v` option.

31.3 OEM Network Data Record

Radisys defined the OEM Network Data Record as a storage for network configuration parameters for the FP eth2, FP eth3, BP eth0, and BP eth1 ports located on each RSM. The OEM record is similar in format to the Shelf Manager IP Connection Record, but with more fields to accommodate all of the eth0, eth1, eth2, and eth3 data.

The layout of OEM Network Data Record is shown in Table 60.

Table 60. OEM Network Data Record

Offset	Length	Definition
0	1	Record Type ID A value of C0h indicates that an OEM record will be used.
1	1	End of List / Version. 7:7 - End of List. Set to 1 for the last record. 6:4 - Reserved. Write as 0. 3:0 - Record format version. Set to 2h for this definition.
2	1	*Record Length*
3	1	*Record Checksum*
4	1	Header Checksum
5	3	Manufacturer ID LS byte first. Radisys Manufacturer ID - 0010F1h will be used.
8	1	Record ID. A value of 0Eh will be used.
9	1	Record Format Version. A value of 00h will be used.
10	1	Port Descriptors. The number of Ethernet ports defined in this record. A value of 8 will be used.
11	4	CMM1 Eth0 IP Address. MS-byte first. Factory default value will be 0.0.0.0.
15	4	CMM1 Eth0 Subnet mask. MS-byte first. Factory default value will be 0.0.0.0.
19	4	CMM1 Eth0 GW. MS byte first. Factory default value will be 0.0.0.0.
23	1	CMM1 Eth0 boot protocol. Factory default value will be 1.
24	4	CMM1 Eth1 IP Address. MS byte first. Factory default value will be 0.0.0.0.
28	4	CMM1 Eth1 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
32	4	CMM1 Eth1 GW. MS byte first. Factory default value will be 0.0.0.0
36	1	CMM1 Eth1 boot protocol. Factory default value will be 1.
37	4	CMM1 Eth2 IP address.MS byte first. Factory default value will be 0.0.0.0.
41	4	CMM1 Eth2 Subnet mask. MS byte first. Factory default will be 0.0.0.0.
45	4	CMM1 Eth2 GW. MS byte first. Factory default value will be 0.0.0.0.
49	1	CMM1 Eth2 boot protocol. Factory default value will be -1.
50	4	CMM1 Eth3 IP address. MS byte first. Factory default value will be 0.0.0.0.
54	4	CMM1 Eth3 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
58	4	CMM1 Eth3 GW. MS byte first. Factory default value will be 0.0.0.0.
62	1	CMM1 Eth3 boot protocol. Factory default value will be -1.
63	4	CMM2 Eth0 IP address. MS byte first. Factory default value will be 0.0.0.0.
67	4	CMM2 Eth0 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
71	4	CMM2 Eth0 GW. MS byte first. Factory default value will be 0.0.0.0.
75	1	CMM1 Eth0 boot protocol. Factory default value will be 1.
76	4	CMM2 Eth1 IP address. MS byte first. Factory default value will be 0.0.0.0.
80	4	CMM2 Eth1 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
84	4	CMM2 Eth1 GW. MS byte first. Factory default value will be 0.0.0.0.
88	1	CMM2 Eth1 boot protocol. Factory default value will be 1.
89	4	CMM2 Eth2 IP address. MS byte first. Factory default value will be 0.0.0.0.
93	4	CMM2 Eth2 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
97	4	CMM2 Eth2 GW. MS byte first. Factory default value will be 0.0.0.0.
101	1	CMM2 Eth2 boot protocol. Factory default value will be -1.
102	4	CMM2 Eth3 IP address. MS byte first. Factory default value will be 0.0.0.0.

Offset	Length	Definition
106	4	CMM2 Eth3 Subnet mask. MS byte first. Factory default value will be 0.0.0.0.
110	4	CMM2 Eth3 GW. MS byte first. Factory default value will be 0.0.0.0.
114	1	CMM2 Eth3 boot protocol. Factory default value will be -1.

31.4 Startup Behavior

The OS network stack of the RSM is initialized as part of the OS load before RSM software stack initialization. At this first network stack initialization, the network data from the Chassis Data Module is not available. This initial start of the OS network stack uses the factory default configuration in the `/etc/sysconfig/network-scripts/ifcfg-ethx` and `/etc/cmm/networks.conf` files. After the RSM has read the network data from the Chassis Data Module as part of the initialization of its software stack, the OS network stack may be reinitialized later.

By default, the RSM assigns IP addresses statically.

- FP eth2, labeled “1” on the front panel, is configured with the static IP address 10.90.91.93
- FP eth3, labeled “2” on the front panel, is configured with a static IP address of 192.168.101.94
- BP eth0 on the backplane is configured with the static IP address 10.90.90.91
- BP eth1 on the backplane is configured with a static IP address of 192.168.100.92
- eth1:1, an alias of eth1 is used to always point to and be active on the active RSM, is configured with a static IP address of 192.168.100.93

On initial power-up of a chassis with two RSMs, both RSMs will have the same IP addresses assigned by default. During election the standby RSM automatically decrements its IP address by one if it detects an address conflict with the active RSM.

Example:

1. Chassis with two (redundant) RSMs is powered up.
2. Active RSM assigns IP address to eth1 of 192.168.101.94.
3. Standby RSM assigns IP address to eth1 of 192.168.101.93.

Note: It is recommended that both RSMs use static IP addresses for all interfaces. DHCP addresses may be unexpectedly lost or changed in some network configurations.

Caution:

- Make sure that the two RSMs do *not* contain duplicate IP addresses on any interface (eth0, eth1, eth2, eth3) to avoid address conflicts on the network.
- Each ethx interface should *always* be assigned to a different subnet. Setting ethx interfaces on the same subnet will cause network errors on the RSM and redundancy will be lost.

31.5 Setting and accessing network configuration data

The proper method to set the network configuration data in the Shelf FRU (after initialization using the FRU update utility) and in `networks.conf` and `/etc/sysconfig/network-scripts/ifcfg-ethxf` configuration files is to use one of the system management interfaces: CLI, SNMP, or ShM API. You can also get the network configuration data through these same interfaces.

Network configuration information for the active RSM can also be set using RMCP.

If the `cmmset` CLI command succeeds, the message `Success` is returned. Otherwise, an error message is returned describing the nature of the error. If the `cmmget` command succeeds, the requested information is returned. Otherwise, an error message is returned describing the nature of the error.

You must set or get the data on the active RSM; you cannot set or get data on the standby RSM.

Caution:

- Changing any of the IP address settings and restarting the network could result in connection loss and a failover occurring based on the rules governing redundancy specified in Chapter 10.0, “High Availability” on page 49.
- The manual method of setting network configuration data (e.g. through the `vi` editor) is not supported. You should avoid doing manual modifications as there is no guarantee that the changes will be propagated into the Shelf FRU and OS network stack.

31.5.1 Setting the Active Network Direction

The direction for the active network on the active RSM can be set to use either the backplane Ethernet ports (eth0, eth1) or the front Ethernet ports (eth2, eth3).

These aspects should be considered when setting the active network direction:

- Setting `activenetworkdir` can only be done on the active RSM, and the setting is synced to the standby RSM.
- The active shelf manager IP address is either `eth1:1` or `eth3:1` based on `activenetworkdir`. By default, the active network direction is set to `0` (backplane) in the shelf FRU, so `eth1:1` is the active shelf manager IP interface. If `activenetworkdir` is set to `front` , then `eth3:1` is the active shelf manager IP interface.
- When Ethernet bonding is enabled, `activenetworkdir` cannot be changed. Setting `activenetworkdir` to `front` when bonding is enabled results in an invalid `set data` error. See *Setting Ethernet Bonding* on page 164 for details

To set the active network direction to the backplane ports, enter the following command:

```
cmmset -d activenetworkdir -v backplane
```

To set the active network direction to the front ports, enter this command:

```
cmmset -d activenetworkdir -v front
```

Both commands return this response if the IP direction is set:

```
Success
```

31.5.2 Getting the Active Network Direction

To get the active network direction, enter this command:

```
cmmget -d activenetworkdir
```

The command returns one of these responses:

```
activenetworkdirection: backplane
activenetworkdirection: front
```

31.5.3 Setting Data for Active RSM

To use the CLI to set network configuration data for the active RSM, enter this command:

```
cmmset -d cdmactivenetwork -v ip:<ifaddr>,nm:<mask>,gw:<gtwy>
```

No target is specified when using this command. Dataitem `cdmactivenetwork` always refers to the eth 1:1 interface.

The string *w.x.y.z* denotes an IP address in dotted quad notation. Separate the IP addresses with a single comma and no spaces.

Each IP address is prefixed with a two-character code denoting the purpose of the information provided.

`ip` — IP address of the Ethernet port

nm — network mask (subnet mask)

gw — IP address of default gateway

Valid network data for the active RSM is propagated to the shelf FRU configuration file (/etc/cmm/networks.conf), and the OS network stack (in that order).

Caution: In a valid configuration, a default gateway can be assigned to only one interface on the RSM board.

31.5.4 Retrieving Data for Active RSM

To get network configuration data for the active RSM using the CLI, enter the following command:

```
cmmget -l cmm -d cdmactivenetwork
```

Note: No target is specified when using this command. Dataitem cdmactivenetwork always refers to the eth 1:1 interface.

31.5.5 Setting Ethernet Port Data

To use the CLI to set network configuration data for Ethernet ports eth0, eth1, eth2, and eth3, enter the following command on the active RSM:

```
cmmset -d cdmcmmNethMdata -v ip:<ifaddr>,nm:<ifmask>,gw:<gtwy>,boot:<boot>
```

No target is specified when using this command.

You can set the port network configuration data for either RSM1 or RSM2 and either eth0, eth1, eth2, or eth3. Specify the RSM to set the data for by replacing *N* with either 1 or 2. Specify the Ethernet port for which to set the data by replacing *M* with either 0, 1, 2, or 3.

The string *w.x.y.z* denotes an IP address in dotted quad notation. Separate the IP addresses with a single comma and no spaces.

Each IP address is prefixed with a two-character code denoting the purpose of the information provided:

ip — IP address of the Ethernet port

nm — network mask

gw — IP address of default gateway

The final prefix indicates the boot protocol:

boot — boot protocol

The value *address_assignment* denotes a value that is either static or dhcp. The value static indicates that the IP address of the port is assigned statically. The value dhcp indicates that the IP address of the port is assigned dynamically using DHCP. Separate address_assignment from the previous values with a single comma and no spaces.

The RSM accepts and stores in both the shelf FRU, and in the networks.conf and ifcfg-ethx files the IP address, network mask, and gateway address specified in the cmmset command even when the boot protocol is specified as dhcp. However, the network stack uses the DHCP protocol to obtain the IP address dynamically. Consequently, using cmmget to retrieve network configuration information returns the data stored in the chassis FRU, not the dynamic IP address assigned to the interface.

Valid Ethernet port data is propagated to the shelf FRU configuration file /etc/cmm/networks.conf (for eth1:1) or /etc/sysconfig/network-scripts/ifcfg-ethx (for other eth interfaces), and the OS network stack (in that order).

31.5.5.1 DHCP Option

eth1:1 always has a static IP address. eth0, eth1, eth2, and eth3 can also be set to use DHCP (Dynamic Host Configuration Protocol) to assign IP addresses. The DHCP client `dhclient` is used instead of `pump`.

A detailed manual page for `dhclient` can be found at:

`http://linux.die.net/man/8/dhclient`

31.5.6 Retrieving Ethernet Port Data

To get network configuration data using the CLI, enter the following command on the active RSM:

`cmmget -l cmm -d cdmcmmNethMdata`

Specify which RSM to get the data for by replacing *N* with either 1 or 2. Specify which Ethernet port for which to get the data by replacing *M* with 0, 1, 2, or 3.

Note: No target is specified when using this command.

31.5.7 Resetting Ethernet Port Data to Factory Default Values

Ethernet port data for eth0 ,eth1,eth2 and eth3 can be reset to factory default values shown in Table 60, “OEM Network Data Record” on page 157 with supplementary tool `clearcdmip`.

Usage is:

`clearcdmip -d cmmNethM`

Specify which RSM to reset the data for by replacing N with either 1 or 2. Specify which Ethernet port for which to reset the data by replacing M with 0, 1, 2, or 3.

31.6 Examples

Here are some examples showing the usage of the cmmget and cmmset commands in the context of IP network configuration.

31.6.1 Setting Active RSM Data

To set the active RSM data, execute the following command:

```
cmmset –l cmm –d cdmactivenetwork
–v ip:10.10.209.91,nm:255.255.255.0,gw:10.10.209.251
```

Response from the cmmset command:

```
Success
```

Retrieve the active RSM data:

```
cmmget –l cmm –d cdmactivenetwork
```

Response from the cmmget command:

```
IPAddress:10.10.209.9
Netmask:255.255.255.0
Gateway:10.10.209.251
```

31.6.2 Setting eth0 Network Configuration Data for RSM1

To set the eth0 network configuration data for RSM1, execute the following command:

```
cmmset –l cmm –d cdmcmm1eth0data
–v ip:10.10.209.91,nm:255.255.255.0,gw:0.0.0.0,boot:static
```

Response from the cmmset command:

```
Success
```

Retrieve the eth0 network configuration data for RSM1:

```
cmmget –l cmm –d cdmcmm1eth0data
```

Response from the cmmget command:

```
IPAddress:10.10.209.91
Netmask:255.255.255.0
Gateway:0.0.0.0
BootProtocol:static
```

31.6.3 Setting eth1 Network Configuration Data for RSM1

To set the eth1 network configuration data for RSM1, execute the following command:

```
cmmset –l cmm –d cdmcmm1eth1data –v
ip:10.10.209.91,nm:255.255.255.0,gw:0.0.0.0,boot:static
```

Response from the cmmset command:

```
Success
```

Retrieve the eth1 network configuration data for RSM1:

```
cmmget –l cmm –d cdmcmm1eth1data
```

Response from the cmmget command:

```
IPAddress:10.10.209.91
Netmask:255.255.255.0
Gateway:0.0.0.0
BootProtocol:static
```

31.6.4 Setting eth2 Network Configuration Data for RSM1

To set the eth2 network configuration data for RSM1, execute the following command:

```
cmmset –l cmm –d cdmcmm1eth2data –v
ip:10.10.209.91,nm:255.255.255.0,gw:0.0.0.0,boot:static
```

Response from the cmmset command:

```
Success
```

Retrieve the eth2 network configuration data for RSM1:

```
cmmget –l cmm –d cdmcmm1eth2data
```

Response from the cmmget command:

```
IPAddress:10.10.209.91
Netmask:255.255.255.0
Gateway:0.0.0.0
BootProtocol:static
```

31.6.5 Setting eth3 Network Configuration Data for RSM1

To set the eth3 network configuration data for RSM1, execute the following command:

```
cmmset –l cmm –d cdmcmm1eth3data –v
ip:10.10.209.91,nm:255.255.255.0,gw:0.0.0.0,boot:static
```

Response from the cmmset command:

```
Success
```

Retrieve the eth3 network configuration data for RSM1:

```
cmmget –l cmm –d cdmcmm1eth3data
```

Response from the cmmget command:

```
IPAddress:10.10.209.91
Netmask:255.255.255.0
Gateway:0.0.0.0
BootProtocol:static
```

31.6.6 Querying Factory Defaults

To query the factory defaults in the Shelf FRU on the chassis, execute the following command:

```
cmmget –l cmm –d cdmactivenetwork
```

Response from the cmmget command:

```
IPAddress: 0.0.0.0
Netmask: 0.0.0.0
Gateway: 0.0.0.0
```

This example assumes you have not yet set the network configuration data and that the Shelf FRU supports storing all the network configuration data.

31.7 Using ShM API to Set and Get Network Configuration Data

You can use the ShM API interface to set and get network configuration data. For details, refer to the "A6K-RSM-J, MPCMM0001 and MPCMM0002 Chassis Management Module ShM & OAM API Reference Manual".

31.8 Using SNMP to Set and Get Network Configuration Data

MIB objects have been defined under the "cmm" group to allow you to use the SNMP Set and Get commands to set and retrieve network configuration data. The objects defined in the MIB correspond to the data items and values defined for the CLI cmmset and cmmget commands.

31.9 Start-up Network Configuration Data

When the operating system boots, the network configuration data present in /etc/sysconfig/network-scripts/template.ifcfg-ethx is copied over to the corresponding /etc/sysconfig/network-scripts/ifcfg-ethx file and the initial values for the network configuration data are taken from the /etc/sysconfig/network-scripts/ifcfg-ethx file.

Once the RSM firmware has booted, the network configuration data is read from the shelf FRU.

If the RSM firmware reads an IP address of 0.0.0.0 for an interface, or if it cannot read and validate the data in the shelf FRU for an interface, the network configuration data for that interface in the /etc/sysconfig/network-scripts/ifcfg-ethx file is used instead. The x in the file name can be 0, 1, 2, or 3.

31.10 Synchronization Between RSMs

The network data synchronized from the active RSM to the standby RSM includes the eth1:1 network details and the eth0, eth1, eth2, and eth3 IP addresses. The standby RSM uses the eth1:1, eth0, eth1, eth2, and eth3 IP addresses to update network.conf and ifcfg-ethx.

31.11 Setting Ethernet Bonding

Ethernet bonding provides high Ethernet availability. Once bonding is activated, the RSM treats the eth0 and eth1 interfaces as a single interface (bond0). If one of the wires from the interface is pulled out and the link goes down, the packets for that interface go through the other one.

Note:

- Only the backplane Ethernet interfaces (eth0 and eth1) support bonding.
- The default setting for bonding is OFF when a new image boots up. This setting is configured in the /etc/cmm/shm.conf file.

31.11.1 Enabling / Disabling Ethernet Bonding

Bonding should be enabled and disabled by setting the BONDING_STATUS variable on both RSMs and then rebooting both RSMs.

31.11.1.1 Enabling

1. From the active RSM, determine the active network direction.

   ```
   cmmget –l cmm –d cdmactivenetwork
   ```

   If the network direction is Front, set the direction to backplane.

   ```
   cmmget –l cmm –d cdmactivenetwork
   ```

   Note: It is not recommended to change the IP address of eth0 and eth1 when bonding is enabled. To change the IP address, restart the RSM after setting the new address.
2. Modify the value of variable BONDING_STATUS to **1** in the /etc/cmm/shm.conf file for both RSMs. By default, the value for BONDING_STATUS is **0** (OFF).
3. Reboot both RSMs. The RSM will come up with bonding enabled.

When bonding is enabled, the active network direction cannot be changed and the network direction is always backplane. Setting activenetworkdir to front when bonding is enabled results in an invalid set data error. See *Setting the Active Network Direction* on page 159 for details about configuring activenetworkdir.

31.11.1.2 Disabling

1. Modify the value of variable BONDING_STATUS to **0** in /etc/cmm/shm.conf for both RSMs.
2. Reboot both RSMs. The RSM will come up with bonding disabled.

31.11.1.3 Enabling / Disabling Bonding While the RSM is Running

Bonding can be manually started, stopped or restarted while the RSM is running by executing the cmmbonding script, as shown in the following example.

```
/etc/init.d/cmmbonding {start | stop | restart}
```

Warning: Starting or stopping bonding using the bonding script may result in unexpected RSM behavior because the ShMgr software may not properly handle manual changes.

31.11.2 Bonding Configuration

- Bonding is enabled in active-backup mode.
- bond0 takes the eth0 IP configuration.
- bond0:2 takes the eth1 IP configuration
- bond0:1 takes the active network IP configuration. Since bonding is available only if the active network direction is backplane, bond0:1 takes the configuration of eth1:1.
- For RSM1, eth0 is the active interface.
- For RSM2, eth1 is the active interface.
- File cmmbonding.conf contains the default bonding values. To change parameters, modify cmmbonding.conf and reboot both RSMs to load the changed parameters.

31.11.3 Verifying Proper Bonding Operation

1. Check if the bonding module is loaded.

   ```
   lsmod | grep bonding
   bonding     96228 0
   ```

2. Check if bonding is running.

   ```
   cat /proc/net/bonding/bond0
   ```

   Output similar to the following displays.

   ```
   Ethernet Channel Bonding Driver: v3.3.0 (June 10, 2008)
   Bonding Mode: fault-tolerance (active-backup)
   Primary Slave: eth0
   Currently Active Slave: eth0
   MII Status: up
   MII Polling Interval (ms): 100
   Up Delay (ms): 100
   Down Delay (ms): 100
   Slave Interface: eth0
   MII Status: up
   Link Failure Count: 1
   Permanent HW addr: 00:00:50:6b:4b:30
   Slave Interface: eth1
   MII Status: up
   Link Failure Count: 0
   Permanent HW addr: 00:00:50:6b:4b:31
   ```

3. Check ifconfig.

   ```
   ifconfig bond0
   ```

   Output similar to the following displays.

   ```
   Bond0 Link encap:Ethernet HWaddr 00:00:50:6B:4B:30
      inet addr:128.0.10.89 Bcast:128.0.10.255 Mask:255.255.255.0
      inet6 addr: fe80::200:50ff:fe6b:4b30/64 Scope:Link
      UP BROADCAST RUNNING MASTER MULTICAST MTU:1500 Metric:1
      RX packets:10182543 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1054934 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:881243726 (840.4 MiB) TX bytes:93801752 (89.4 MiB)
   ```

   ```
   ifconfig bond0:2
   ```

   Output similar to the following displays.

   ```
   Bond0:2 Link encap:Ethernet HWaddr 00:00:50:6B:4B:30
      inet addr:128.0.10.151 Bcast:128.0.10.255 Mask:255.255.255.0
      inet6 addr: fe80::200:50ff:fe6b:4b30/64 Scope:Link
      UP BROADCAST RUNNING MASTER MULTICAST MTU:1500 Metric:1
   ```

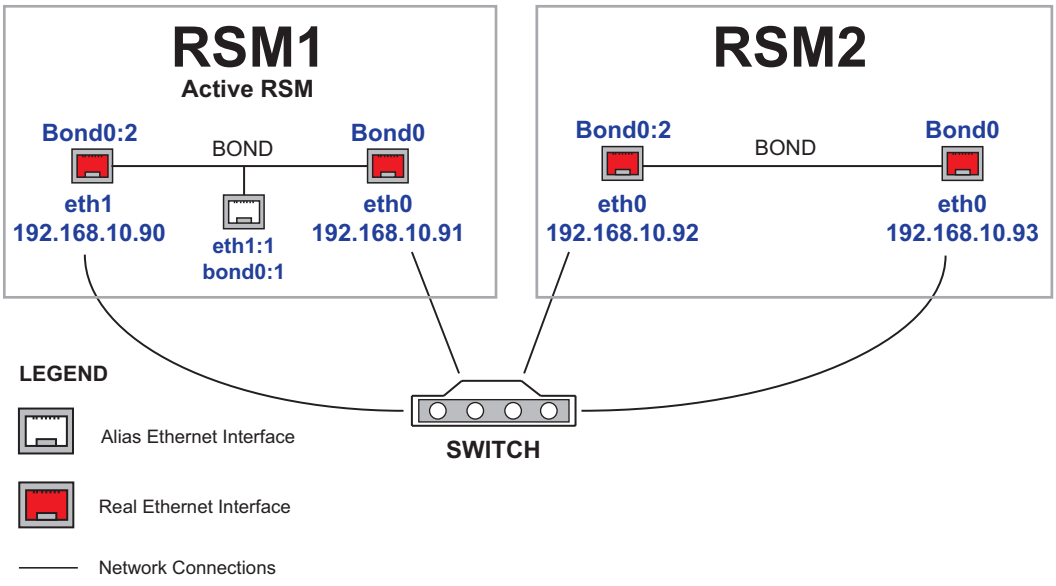
31.11.4 Bonding Tests

These basic checks can be done to test Ethernet bonding:

- Check if the **`ifconfig`** command returns bonding interface details.
- Check for an active bonding interface.
- Remove the cables for either eth0 or eth1 for an RSM, then check if there is connectivity.
- Perform a failover and check if the active bonding interface is operational.

Follow these steps to verify high availability of the RSM interfaces through bonding of eth0 and eth1. Refer to the following diagram for details.

1. Pull the eth0 cable for RSM1 and check for connectivity.
2. Check the current active slave (refer to the terminal output in the following diagram).
3. Similarly, pull the eth1 cable in RSM2 and check the active slave.



32.0 Updating RSM Software

32.1 Overview

The RSM is capable of having its firmware and critical system files updated when new update packages become available. The update process allows these updates to occur remotely without losing the active RSM in a redundant configuration.

When new RSM updates are available, they are packaged in a .tgz file. See the *A6K-RSM-J Shelf Manager Firmware and Software Update Instructions* for details on performing the updates.

32.2 Main Features of Firmware Update Process

The main features of the firmware update process are:

- Updates can be done remotely over the front or back Ethernet ports on the RSM
- Dual Image provides redundant storage for firmware images.
- Current RSM configuration data is preserved across the update
- Critical RSM data such as the SEL and command history is preserved across an update
- Redundant RSMs can be updated without interrupting management of the chassis
- Update files are verified and checked for corruption
- Update components have associated version numbers
- Update events are logged to the SEL
- Updates can be triggered using the CLI
- Update packages can be located locally on the RSM or pulled from a mounted NFS, remote FTP or TFTP server.

32.3 Update Process Elements

The RSM update process relies on the following elements:

- User Client – The client triggers the update process, and can be located anywhere on the network. The CLI interface on the RSM can be used to trigger the firmware upgrade.
- Update Package – The update package contains the new software components and other files necessary for the update. The update package can be pulled from a remote server, or be pushed locally onto the RSM.
- RSM Upgrade Manager – This is an RSM software entity that processes incoming update requests and responses to them over the various interfaces exposed by the RSM.
- Update Package Server (Optional) – The update package server can store update packages remotely from the RSM. This can be an NFS, FTP, or TFTP server.

32.4 Dual Image

The RSM update process uses a dual-image scheme to manage all local images. The scheme assumes that two instances of images are kept in separate flash memory chips. The active flash chip is the chip containing the code that is currently running. The inactive, or backup, flash chip is the location where the new image is loaded.

32.4.1 Next Boot Role

The role for each image set can be selected at any time. The role determines which image will be active after the device restarts. Table 61, “Image Set Next Boot Roles” lists what image roles are available.

Table 61. Image Set Next Boot Roles

Next Boot Role	Description
DEFAULT(0)	The image set will be used to boot the system, assuming that all components are validated correctly.
FALLBACK(1)	The image set will be used to boot the system if any image in the active set is broken.

Configured image set next boot roles are written into the non-volatile memory. Table 62, “Allowed Next Boot Role Combinations” lists the allowed combinations.

Table 62. Allowed Next Boot Role Combinations

Image Set 1 Next Boot Role	Image Set 2 Next Boot Role
DEFAULT	INACTIVE
INACTIVE	DEFAULT
DEFAULT	FALLBACK
FALLBACK	DEFAULT

After a successful next boot role change operation, an event is posted into the SEL.

32.4.2 Setting the Next Boot Role

The next boot role for a specific image set can be set using the CLI command:

```
cmmset –t image:<type>:<instance> -d NextBootRole –v <role>
```

Table 63. Setting the Next Boot Role - Command Options

type	(mandatory) Image type. Allowed values: “All images”
instance	(mandatory) image set instance. Allowed values: 0, 1
role	(mandatory) Specifies the image next boot role. Possible values: • default • fallback

The command returns an error if the selected <role> leads to an invalid combination.

32.4.3 Automatic Rollback

If the image does not work properly, the system can be restarted using a CLI command. It may also happen that the system hangs and is restarted by the watchdog hardware. In both cases, automatic rollback of the upgrade procedure is performed. When the system starts after an unsuccessful upgrade, it will use the system from the partition containing the old image. The status of the partition containing the old image will be restored to DEFAULT. Additionally, an event using the upgrade sensor is posted to the SEL indicating the unsuccessful upgrade.

32.4.4 System Booting Failures

The system may detect that both partitions contain at least one image with a broken checksum. In this case, the booting procedure is terminated, the system displays an error message, and waits for commands from the user. The boot loader makes it possible to upgrade an arbitrarily selected partition using the Xmodem protocol. It also makes it possible to set the proper image status word value to enable the system to boot from the new image. The functionality is also useful when the boot loader detects an illegal value of Image Status Word.

After an unsuccessful upgrade, the upgraded partition contains the broken image. In such a case, the system might not boot when the old image on the active partition is broken. If the system boots to U-Boot, it will wait for user requests as described in Section 32.14, “U-Boot Update Process” on page 174.

32.4.5 Restarting Specified Image

A specific image may be restarted using the CLI command:

```
cmmset –t image:<type>:<instance> -d restart –v 1
```

Table 64. Restarting a Specified Image - Command Options

type	(mandatory) image type name. Allowed values: • OS loader • Root filesystem • Linux kernel • NAND FPGA • All images
instance	(mandatory) image instance. Allowed values: 0, 1

32.5 Critical Software Update Files and Directories

Table 65, “List of Critical Software Update Files and Directories” lists files and directories important to the RSM update process.

Table 65. List of Critical Software Update Files and Directories

File or Directory Name:	Description:
/tmp/upgradeXXXXX	Temporary directory into which the update package is copied and unzipped. The update process will delete and recreate this directory. X is a random alphanumeric character.
[package file].tgz	Archive file containing update package files

32.6 Generating the update package

The RSM update bundle file is provided as *CMM3-upd-<version>.tgz*. A script file must be extracted from the bundle, then executing the script file generates the *install.tgz* update package required by the update process.

Follow this procedure to generate the required *install.tgz* update package.

1. Download *CMM3-upd-<version>.tgz* to the directory where the update process will be invoked.
2. Extract script *transform.sh* from the update bundle.

   `tar zxf CMM3-upd-<version>.tgz transform.sh`

3. Run *transform.sh* on the update bundle to generate the *install.tgz* update package.

   `/transform.sh CMM3-upd-<version>.tgz`

Use *install.tgz* to update the RSM. See the *A6K-RSM-J Firmware and Software Update Instructions* for details about the update process.

32.7 Update Package

The *install.tgz* update package contains the components listed in Table 66, “Contents of the Update Package”.

Table 66. Contents of the Update Package

Update File	Description
cmm3_all.hpm	IPMI firmware
u-boot-spi.bin	U-Boot image
Linux.bin	Linux and ShMgr software images

The update package can be placed locally on the RSM in the user specified directory, or it can reside on a server on the network.

Arguments for the location of the update package can be given in the CLI command. It is here that you can point to a remote server or a local directory.

Note: If an NFS server is mounted to the RSM, the argument in the update script will be similar to a file located locally on the RSM.

If the package fails to copy or transfer to `/tmp/upgradeXXXXX`, the update process will terminate.

32.7.1 Update Package File Validation

The procedure starts with verification of the checksum of the package meta-data file containing the package contents description. Next, the verification procedure checks the following data for each of the images to be upgraded:

- Image Header Checksum
- Image Checksum
- Target Platform Indicator
- Image Size – the Upgrade Manager checks whether the image fits the target partition size
- Image Version – the Upgrade Manager checks whether the new image version is different than the old image version unless FORCE install is requested

At any time, validation of all installed packages can be done using this CLI command:

```
cmmget -d verifyImages
```

32.7.2 Firmware Image Properties

The installed firmware images have a number of properties associated with them. The properties for the installed firmware image can be retrieved using the CLI command:

```
cmmget –t image:<type>:<instance> -d properties
```

Table 67. Firmware Image Properties - Command Options

type	(mandatory) image type name. Allowed values: • OS loader • Linux kernel • Root filesystem • NAND FPGA • All images
instance	(mandatory) image instance. Allowed values: 0, 1

32.8 Single RSM System

In systems with a single RSM, the update procedure is done on the active RSM that controls the shelf operation. The image update does not require RSM shutdown, but a restart is required to boot from the upgraded image set.

32.9 Redundant RSM Systems

In systems with redundant RSMs, the update can only be done on the standby RSM. After the update is complete, initiate a failover from the active to the standby and update the second RSM which is now the standby.

32.10 CLI Software Update Procedure

The CLI supports a command for an update request. The syntax of the command is as follows:

```
cmmset –d update –v [image] [option] [ftp:server:user:password]
```

To update UBoot, Linux, the shelf manager software and the IPMC on an RSM with one invocation of `cmmset`, follow the syntax in this example command:

```
cmmset –d update –v "/tmp/install ipmc yesact"
```

Table 68. CLI software update - command options

image	(mandatory) The pathname (including the file name) of the update package file without the .tgz extension. For example: /usr/local/cmm/temp/CMM
ftp	(optional) The final set of arguments is used if the update package is located on a remote FTP server. If ftp is supplied as an argument, the server and user arguments are also required. The password argument is optional, but if it is not supplied, then FTP server will prompt for a password during the establishment of the FTP connection. ***ftp***—Optional argument used to indicate that the update package resides on a remote FTP server. If this argument is supplied, the arguments for server and user must also be supplied. The argument for password is optional. ***server***—Argument that gives the hostname or IP address of the FTP server where the firmware update package is stored. ***user***—Argument that provides the username to be supplied to the FTP server for authentication. ***password***—Optional argument that is supplied to the FTP server for authentication. For example: cmmset -d update -v "/upgrade/CMM/install ftp:192.168.1.1:username:password"

Note: The -v argument can be up to 128 characters long.

The command returns a 0 if the update request is successful, and non-zero if an error occurs.

32.11 Update Process

1. The client initiates an update request via a CLI command
2. The RSM validates the update request
   - — The RSM is not already doing an update
   - — In a redundant configuration, the RSM must be standby
3. If the update request is valid then
   - — Continue
4. Else
   - — Exit
5. If FTP arguments are supplied then
   - — Retrieve the package file from the FTP server to the `/tmp/upgradeXXXXX` directory
   - — Exit if an error occurs
6. Unzip the .zip file in the `/tmp/upgradeXXXXX` directory
7. Validate the checksum for all files in the unzipped package
   - — Exit if any files fail
8. Validate the image length for all files in the unzipped package
   - — Exit if any files fail
9. Validate that all files in the unzipped package match the RSM platform ("atca")
   - — Exit if there is a mismatch
10. Write images on the flash memory location for each image included in unzipped package
    - — Erase the flash partition for the given image
    - — Write the new image on the flash partition

— If a component update fails:

a. Stop updating components

b. Exit the update process, but do not reboot

11. If the process has been successful so far then

a. Set the image boot role for the image that was updated:

— DEFAULT, otherwise

b. Set the image boot role for the image set that was the active one during the update procedure:

— FALLBACK

c. Reboot the RSM. Reboot is not performed by the upgrade procedure, so a separate user command is required.

32.12 Local Upgrade Sensor

Upgrade Manager uses the "Local Upgrade" Sensor to provide information on the status of the RSM update process. This is an event-only sensor that cannot be queried through system management interfaces. For a detailed description refer to Appendix D, “OEM Sensor Events”.

32.13 Configuration Upgrade

An RSM configuration upgrade is based on the following assumptions:

- All RSM configuration files keep configuration data in form of <keyword, value> pairs.
- When an RSM module encounters an unknown keyword in a configuration, it skips the parameter.
- When a RSM module encounters a keyword with an illegal value, or the configuration file does not contain the keyword, the module applies a default value for the parameter.

There is no need to convert the configuration files during the RSM image upgrade because the RSM modules can run using the old configuration files[1]. They skip unused parameters and use default values for new parameters.

32.14 U-Boot Update Process

The firmware can also be updated through U-Boot. This update is done at a pre-OS level, meaning that the update is executed before the OS loads. This method requires updating over TFTP through the eth0 Ethernet port and must be done locally. A separate update package is needed if this method is used. The instructions are included with the update package.

Because this process can completely erase the flash and operates in a pre-OS environment, it can be used as a failsafe to recover from failed firmware updates done from the command line interface.

1. This does not hold for heterogeneous upgrades.

Chapter 33

33.0 Chassis Component Firmware Update

Certain devices in the chassis that are managed with an IPMC (Intelligent Platform Management Controller) can have their FRU information and firmware updated either locally or remotely through the RSM. Devices in the chassis that can potentially be updated include the CDMs, the fan trays, and the PEMs. The RSM can also potentially be used to update firmware on blades in the chassis.

Instructions on updating devices in a chassis (including the CDMs, PEMs, and fan trays) can be found in the documentation for the specific chassis.

For instructions on updating the firmware on the A6K-RSM-J shelf manager, see the *A6K-RSM-J Shelf Manager Firmware and Software Update Instructions*.

Documentation and firmware for products designed for AdvancedTCA specifications from Radisys can be found in the downloads section at http://www.radisys.com.

Chapter 34

34.0 FRU Update Utility

34.1 Overview

The `fru_update` shell script can be used for two purposes:

- To update the portions of the functional FRU data that changed to a new version from Radisys while preserving FRU-specific information.
- To modify certain customizable fields in the FRU data while preserving the functional FRU data.

34.2 FRU Update Architecture

The `fru_update` script reads the existing FRU data from the FRU device, then creates a new FRU image that combines the existing FRU data with the data to be modified. A configuration file indicates the parts to be modified. The new image is then written to the FRU device. A copy of the original FRU image is saved temporarily and then removed once the update has completed successfully.

The `fru_update` script uses the `frutool` and `rsys-ipmitool` executables. The `fru_update` and `frutool` utilities verify the files to be used in advance, and also verify the data contained in the device after the update.

34.2.1 Required Files

These files are required to complete the FRU update:

- `fru_update` BASH script
- `rsys-ipmitool` and `frutool` executables. These applications must be present in the PATH environment variable.
- One of these pairs of files:
  - Files from Radisys with names ending in <version>.cfg and <version>.bin to use for upgrading the functional FRU information. Do not modify or compile these files before use.
  - Files with names ending in CustomFields.cfg and CustomFields.bin that are modified with custom data.

For each Radisys FRU information device, there are two pairs of FRU update files. One set is a versioned .cfg and .bin pair which are used for upgrading functional FRU information. This procedure is described in *FRU Update Usage* on page 177.

The second set is a pair of .cfg and .sf files marked as being for Custom Fields, which can be used to modify customer specific fields. The use of these is described in *Customizing FRU-Specific Data* on page 181.

34.2.2 Update Verification

There are many checks present in both the `fru_update` script and `frutool` to ensure that errors cannot occur when updating the device FRU information. These are the verification tasks:

- Verify the .cfg and .bin files are a matching pair
- Verify the .cfg file is complete and correct
- Verify the target device and .cfg/.bin files match
- Verify the data integrity of the device FRU data and update .bin files
- Verify the data written back to the device matches what it should be

34.2.3 FRU Data Recovery

If a FRU data area becomes corrupted during an update, the update cannot be forced because `fru_update` cannot decide what data is supposed to be there or what data is actually valid or invalid. Consequently, manual intervention is required to recover the original FRU data.

When `fru_update` is run, it creates backup copies of the FRU data in the current working directory. The FRU backups can be used with `rsys-ipmitool` to restore the data if the RSM is reset or loses power during the upgrade or downgrade.

Invoke `fru_update` from a head machine where the backup copies will not be lost, or from a directory on the RSM that is in persistent storage. If `fru_update` is to be invoked from the RSM LMP, change the working directory to a directory mounted on the JFFS2 file system so the FRU backup copy is not lost.

34.2.3.1 Shelf FRU Backup Commands

The shelf FRU data is stored in files `shelffru1.bin` and `shelffru2.bin`. To create a backup of the shelf FRU data, use the `rsys-ipmitool` utility.

Caution: The files `shelffru1.bin` and `shelffru2.bin` should be backed up on a non-volatile storage device, such as a head system hard drive, so the files are not lost during an LMP reset or upgrade.

Use the following commands to create a backup copy of the shelf FRU data. For this example, the left RSM in the chassis is called RSM1, and the right RSM in the chassis is called RSM2.

If you are operating on RSM1 (left):

```
rsys-ipmitool -t 0x20 -m 0x10 fru read 1 shelffru1.bin
rsys-ipmitool -t 0x20 -m 0x10 fru read 2 shelffru2.bin
```

If you are operating on RSM2 (right)

```
rsys-ipmitool -t 0x20 -m 0x12 fru read 1 shelffru1.bin
rsys-ipmitool -t 0x20 -m 0x12 fru read 2 shelffru2.bin
```

34.2.3.2 Shelf FRU Recovery Command

To restore the previous shelf FRU data after corruption has occurred, invoke the `rsys-ipmitool` utility from the head machine or persistent storage area where the backup shelf FRU data was saved. Specify the name of the backup FRU `.bin` file. This is an example command:

```
rsys-ipmitool -m 0x12 -t 0x20 fru write 2 shelffru1.bin
```

34.3 FRU Update Usage

This is the command syntax for the `fru_update` utility.

```
fru_update "<ipmitool params>" <update cfg> <fru image>
```

`<ipmitool params>` are the ipmitool parameters to access the device. See *ipmitool Parameters* for a complete list.

The IPMB address of the chassis slot or FRU is needed for some ipmitool parameters. See *Chassis slot and FRU IPMB addresses* for a list of addresses.

`<update cfg>` is the name of the FRU update configuration file (`<filename>.cfg`)

`<fru image>` is the latest binary FRU data file (`<filename>.bin`)

Note: Invoke `fru_update` from a directory on the RSM that is persistent storage. The utility creates a backup of the current FRU data in the working directory so the FRU data can be recovered if the update fails or data corruption occurs. See *FRU Data Recovery* for details.

34.3.1 ipmitool Parameters

The `ipmitool` parameters are listed in the following table. The information in this table can also be displayed by invoking **`ipmitool --h`**. Only some of the parameters are used with `fru_update`.

Table 69. ipmitool Parameters Available to fru_update (Sheet 1 of 2)

Parameter	Description
-h	This help information
-V	Show version information
-v	Verbose (can use multiple times)
-c	Display output in comma separated format
-d N	Specify a /dev/ipmiN device to use (default=0)
-I intf	Interface to use
-H hostname	Remote host name for LAN interface
-p port	Remote RMCP port [default=623]
-U username	Remote session username
-f file	Read remote session password from file
-S sdr	Use local file for remote SDR cache
-a	Prompt for remote password
-e char	Set SOL escape character
-C ciphersuite	Cipher suite to be used by lanplus interface
-k key	Use Kg key for IPMIv2 authentication
-L level	Remote session privilege level [default=ADMINISTRATOR] Append a '+' to use name/privilege lookup in RAKP1
-A authtype	Force use of auth type NONE, PASSWORD, MD2, MD5 or OEM
-P password	Remote session password
-E	Read password from IPMI_PASSWORD environment variable
-m address	Set local IPMB address
-b channel	Set destination channel for bridged request
-t address	Bridge request to remote target address
-B channel	Set transit channel for bridged request (dual bridge)
-T address	Set transit address for bridge request (dual bridge)
-l lun	Set destination lun for raw commands
-o oemtype	Setup for OEM (use 'list' to see available OEM types)
-O seloem	Use file for OEM SEL event descriptions
Interfaces	
lan	IPMI v1.5 LAN Interface [default]
lanplus	IPMI v2.0 RMCP+ LAN Interface
Commands	
raw	Send a RAW IPMI request and print response
i2c	Send an I2C master write-read command and print response
spd	Print SPD info from remote I2C device
lan	Configure LAN channels
chassis	Get chassis status and set power state
power	Shortcut to chassis power commands
event	Send pre-defined events to MC

Table 69. ipmitool Parameters Available to fru_update (Sheet 2 of 2)

Parameter	Description
mc	Management Controller status and global enables
sdr	Print Sensor Data Repository entries and readings
sensor	Print detailed sensor information
fru	Print built-in FRU and scan SDR for FRU locators
sel	Print System Event Log (SEL)
pef	Configure Platform Event Filtering (PEF)
sol	Configure and connect IPMIv2.0 Serial-over-LAN
tsol	Configure and connect with Tyan IPMIv1.5 Serial-over-LAN
isol	Configure IPMIv1.5 Serial-over-LAN
user	Configure Management Controller users
channel	Configure Management Controller channels
session	Print session information
sunoem	OEM commands for Sun servers
kontronoem	OEM commands for Kontron devices
picmg	Run a PICMG/ATCA extended cmd
fwum	Update IPMC using Kontron OEM Firmware Update Manager
firewall	Configure firmware firewall
exec	Run list of commands from file
set	Set runtime variable for shell and exec
hpm	Update HPM components using PICMG HPM.1 file
check	Check the target information
check <file>	Display the existing target version and image file version on the screen
upgrade <file>	Upgrade the firmware using a valid HPM.1 image <file>
upgrade <file> all	Updates all the components present in the <file> regardless of version numbers (use this only after "check" command)
upgrade <file> component x	Upgrade only component <x> from the given <file> component 0 - boot component 1 - application component 2 - FPGA IPMC component 3 - FPGA Fawkes
upgrade <file> activate	Upgrade the firmware using a valid HPM.1 image <file>. If activate is specified, the IPMI controller will reset and use the newly uploaded image.
activate	Activate the newly uploaded firmware
rollback	Causes the active application image to become the backup and the backup image to become active. ***Note:*** This should be used with caution because the backup image may not be compatible with other components.
noprompt	Suppresses messages or prompts generated by the utility

34.3.2 Chassis slot and FRU IPMB addresses

This section lists the slot and FRU IPMB addresses for each supported chassis type. The IPMB address is required when the `-m` option is used with the `fru_update` and `rsys-ipmitool` utilities.

Table 70. Chassis slot and FRU IPMB addresses

Chassis slot or FRU	IPMB address (hex)					
	Schroff 2-slot (11596-099)	NECCH0001	ATCA-6014 10G	ATCA-6014 40G	Schroff 14U (11596-008)	Schroff 14U (11596-151)
1	82	9A				
2	84	96				
3	n/a	92				
4	n/a	8E				
5	n/a	8A				
6	n/a	86				
7	n/a	82				
8	n/a	84				
9	n/a	88				
10	n/a	8C				
11	n/a	90				
12	n/a	94				
13	n/a	98				
14	n/a	9C				
PEM 1 (left from rear)	n/a	60, FRU ID 6				
PEM 2 (right from rear)	n/a	60, FRU ID 7				
Fan 1 (viewed from front)	n/a	60, FRU ID 3/Left fan tray				
Fan 2 (viewed from front)	n/a	60, FRU ID 4/Center fan tray				
Fan 3 (viewed from front)	n/a	60, FRU ID 5/Right fan tray				
RSM 1 (left)	10					
RSM 2 (right)	12					
Active shelf manager	20					

34.3.3 Command Examples:

The following command is run on the RSM in the left slot of a two-slot chassis (slot address 0x10). An OpenIPMI connection is made and the utility targets address 0x20 on the IPMB.

```
fru_update "-t 0x20 -m 0x10" <version>.cfg <version>.bin
```

This command is run on the RSM in the right slot of a two-slot chassis (slot address 0x12):

```
fru_update "-t 0x20 -m 0x12" <version>.cfg <version>.bin
```

The scripts verify the type of FRU being updated against the files provided before writing the data.

34.4 Customizing FRU-Specific Data

The frugen.pl PERL script prompts for new values for the user-defineable fields in an existing FRU data image. The script creates a new binary image containing the functional FRU data and the custom values. Specify in a configuration file which of the user-definable fields to overwrite in the FRU device. Use the configuration file and the image created to write the custom values to the FRU device as described in *FRU Update Usage*.

Requirements:

- `frugen.pl` PERL script
- Math::BigInt, Getopt::Long, and Time::Local PERL modules installed
- `fru_update` BASH script
- `frutool` and `rsys-ipmitool` executables in the PATH environment variable on the host where `fru_update` executes
- .cfg and .sf files configured for updating customer defined fields on the desired target device. These are marked as being for 'Custom Fields.'

1. Determine what data will be entered into the customer-defined fields. The following fields are customizable:

   - `Chassis Info Area -` (chassis FRU data only)
     `Chassis Custom 2`
     `Chassis Custom 3`
     `Chassis Custom 4`
   - `Board Info Area -`
     `Board Product Name`
     `Board Part Number`
     `Board Custom 1`
     `Board Custom 2`
     `Board Custom 3`
   - `Product Info Area -`
     `Asset Tag`
     `Product Custom 1`
     `Product Custom 2`
     `Product Custom 3`

2. Compile the custom fields .sf file into a .bin file using `frugen.pl` on a command line:

   ```
   frugen.pl -f <sf_file>.sf -o <bin_file>.bin
   ```

   <bin_file> is the name of the file to be created. Make the <bin_file> base name match the <sf_file> base name.

   The script prompts you to enter a value for each custom field.

3. Respond to the prompts by entering custom data or leaving fields blank to keep the existing value.

   Pressing enter without entering anything uses the data already in the .sf file, which are typically blank spaces, or the data on the FRU device.

   The data entered must match the default length of the field (usually 20 characters). Otherwise, `frugen.pl` prompts again for the same field. Use spaces or other characters to make the input value match the length required.

   The data can also be specified on the command line for scripting purposes. For example:

   ```
   frugen.pl -f <filename>.sf -o <filename>.bin -noi

   -d "Board Product Name"="Custom BrdProdName  "

   -d "Board Part Number"="Custom BrdPartNum   " ... etc.
   ```

An error appears if a -d option for any customizable field is not specified on the command line.

4. Open the custom data .cfg file in a text editor.
5. Uncomment the lines in the file that represent the fields to be overwritten in the FRU device. To uncomment a line, delete the # character and leave no white space at the beginning of the line.

   To keep the existing data that is in the FRU device for a field, keep the # character in front of the field.

   These fields can be uncommented:

   Chassis info area (for shelf FRU data only):

   ```
   #CHASSIS REPLACE CUSTOM 2
   #CHASSIS REPLACE CUSTOM 3
   #CHASSIS REPLACE CUSTOM 4
   ```

   Board info area:

   ```
   #BOARD REPLACE PRODNAME
   #BOARD REPLACE PARTNUM
   #BOARD REPLACE CUSTOM 1
   #BOARD REPLACE CUSTOM 2
   #BOARD REPLACE CUSTOM 3
   ```

   Product info area:

   ```
   #PRODUCT REPLACE ASSETAG
   #PRODUCT REPLACE CUSTOM 1
   #PRODUCT REPLACE CUSTOM 2
   #PRODUCT REPLACE CUSTOM 3
   ```
6. Write the customized fields into the device FRU data with `fru_update`:

   ```
   fru_update "<ipmitool params>" <filename>.cfg <filename>.bin
   ```

   See *FRU Update Usage* on page 177 for details.

Chapter 35

35.0 Third-Party Chassis Integration

35.1 Introduction

The A6K-RSM-J Shelf Manager (RSM) can be integrated into most chassis that comply with the "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". Provided with the proper configuration information, such as IPMB topology, slot layout, hardware addresses, and so on, the RSM firmware is able to manage most third party chassis that have been developed for the RSM hardware according to the RSM hardware specifications and design.

When the RSM initially starts, the startup process reads the chassis FRU to determine manufacturer's name and product name. Based on what it reads from the chassis FRU, the RSM loads specific files and configuration information necessary to access and manage the various elements in the chassis. Chassis configuration files for chassis that are manufactured by Radisys are located in a directory under `/etc/cmm/chassis`. Chassis configuration files for chassis not manufactured by Radisys are located in the same directory.

This chapter describes the steps to create the necessary files and configure the RSM firmware to work in a chassis. You should have a thorough understanding of the "Intelligent Platform Management Interface Specification v1.5", as well as the "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification". Detailed information regarding the information used to create the files necessary for the RSM can be found in these specifications.

35.2 Integrating RSM Firmware into Chassis

The following is a brief outline of the steps necessary to integrate the RSM firmware into a chassis. The steps are discussed in detail in subsequent sections:

1. Create the chassis FRU file as described in Section 35.3, "Creating Chassis FRU Information" on page 183.
2. Install the chassis FRU file into the chassis.
3. Create the configuration files as described in Section 35.4, "Creating Configuration Files" on page 184.
4. Install the new configuration files in the appropriate directory on the RSM.
5. Reboot the chassis.

35.3 Creating Chassis FRU Information

Appropriate FRU information must exist in the chassis for the RSM to function properly. The FRU must follow the appropriate specifications for AdvancedTCA "PICMG 3.0 Revision 2.0 AdvancedTCA Base Specification" as well as be compliant with the "Intelligent Platform Management Interface Specification v1.5".

Chassis FRU information is managed using the `frugen.pl` utility.

35.3.1 About frugen.pl

The `frugen.pl` utility is a PERL script that uses a .sf input file for basic FRU data contents and generates a binary .bin file. The input text file contains the hex data for the FRU.

PERL module requirements: Math::BigInt, Getopt::Long, and Time::Local

35.3.2 Command Options

These are the command line options for `frugen.pl`:

`-f`	Input file name
`-o`	Output file name
`-noi`	non-interactive; no prompt is given for FRU data expected on command line `'-d'`
`-auto`	automated mode, if interactive then no retries are allowed'
`-d`	FRU data, -d "name"="value"
`-p`	pad the entered FRU data with spaces to required length
`-h`	help

Command example:

```
frugen.pl -f <filename>.sf -o <filename>.bin -noi
```

Additional information about the `frugen.pl` utility is available in *Customizing FRU-Specific Data* on page 181.

35.4 Creating Configuration Files

The RSM requires several files to operate in a chassis. These files include information about the chassis and its various components that the RSM needs to manage. All of the files are ASCII files that can be created using any standard text editor.

Chassis configuration files are stored in a directory under the `/etc/cmm/chassis` directory. The chassis configuration directory naming convention is the concatenation of the chassis manufacturer's name and the product name of the chassis as defined in the manufacturer and product name field in the board area of the chassis FRU.

For example, if the manufacturer field in the board area of the chassis FRU contains the value "Acme", and the product name is "ABCD0001", the directory in which to store all of the chassis configuration files is called `/etc/cmm/chassis/ACME_ABCD0001`. See Section 35.6, "Installing Configuration Files" on page 189 for more information about creating the directory and adding the files to the RSM.

Note: The chassis directory name must be in all UPPER CASE letters. Further, the chassis name portion of the chassis directory name can match either the entire chassis name stored in the chassis FRU or just a proper prefix of the chassis name stored in the chassis FRU. In other words, the chassis name stored in the chassis FRU can have "extra" letters (like a suffix) after the chassis name and the directory name will still be treated as a match by the RSM firmware.

File `storage.cfg` is not used. Parameters `Serial` and `chassisMatch` were moved to the RSM configuration file `local.conf`. Location alias to FRU ID mappings were moved to the `cmm.ini` configuration file into section [Alias Output]. All other parameters were deleted as obsolete. Files `*.sif` are not used. The implementation specific information for sensors was integrated into the relevant `[Device`*n*`]` section as the `Sensor`*n* parameter.

35.5 cmm.ini

The `cmm.ini` configuration file on the RSM describes the physical IPMB layout of the chassis and how these physical IPMBs map to logical devices. The `cmm.ini` file must be created for each chassis that the RSM manages.

The `cmm.ini` configuration file is made up of several sections: IPMB, Alias Input, Alias Output, CMM, Blade, FanTray, PEM, Logical Bus, Power Feed, and Fan.

This section also describes any alias information for devices.

35.5.1 IPMB Section

The IPMB section describes the logical device mapping to the devices they are being mapped to. Logical devices correspond to the location argument (as in the command `cmmget -l` *location*) of the various interfaces on the RSM.

The format of the IPMB section is:

`NumLogicalDevs=`*n*

`LogicalDev0=`*device_name*

`...`

`LogicalDev`*n*`=`*device_name*

n: Number of devices (FRUs) connected to the RSM.

device_name: The name of the device connected to a particular LogicalDev*j*. This device name is used later in the file to describe the hardware address and physical bus connected to that logical device.

Note: The `LogicalDev`*n* entries are numbered beginning with 0. This is different from the blade locations in the CLI where numbering of blades begins with 1 (as in `blade1`, `blade2`, and so on).

35.5.2 Alias Input Section

The Alias Input section describes the name of the aliases of logical devices used for input.

The format for the Alias Input section is:

alias_name=logical_device_name

For example, if `blade1` is to be also referred to as `FirstBlade`, you can enter an alias as follows:

```
FirstBlade=blade1
```

You can then use the alias instead of the logical device name. For example, to list all the targets for blade1, you can enter this command:

```
cmmget -l FirstBlade -d listtargets
```

35.5.3 Alias Output Section

The format for this section is:

```
logical_device_name:fru_id=alias_name
```

For example, if chassis:6 is designated as FilterTray1 in the RSM output commands, define the following alias:

```
Chassis:6=FilterTray1
```

With this alias in effect, chassis:6 will be referred as FilterTray1 in the output of all queries (such as `cmmget -l system -d listpresent`).

35.5.4 CMM Section

This section contains the logical bus number and hardware addresses for the primary and secondary physical busses. Since the logical bus between the two RSMs remains fixed and the hardware addresses do not change, this section should remain the same for all implementations.

The format for this section is:

HWAddress0=*hardware address of CMM0*

HWAddress1=*hardware address of CMM1*

35.5.5 Blade Section

The Blade section contains the logical bus numbers and hardware addresses for the primary and secondary buses connecting the RSM to each Single Board Computer (SBC or blade).

The format for this section is:

```
[Blade0]
Address=IPMI_address_of_blade0
[Blade1]
Address=IPMI_address_of_blade1
...
[BladeN-1]
Address=IPMI_address_of_blade(n-1)
```

Note: Blade # starts at 0.

Logical Bus: This is the bus mapped to the physical IPMB connection in the Logical Bus section of the `cmm.ini` file. The logical bus must be assigned a number from 0 to *m*, where *m* is the number of logical busses in the system.

n: Number of blades in the system.

35.5.6 FanTray Section

The Fan Tray section defines the logical bus number and hardware addresses for the primary and secondary buses connecting the RSM to the fan trays.

The format for the section is:

[FanTray1]

Address=*IPMI address of fantray 1*

...

[FanTray*N*]

Address=*IPMI address of fantray n*

n: Number of fan trays in the chassis. The fan tray sections are numbered from 1 though *n*.

35.5.7 PEM Section

The PEM section defines the logical bus and hardware address information for connecting the RSM to the Power Entry Modules (PEMs). The format for the section is as follows:

[PEM0]

Address=*IPMI address of PEM 0*

...

[PEM*n-1*]

Address=*IPMI address of PEM n-1*

n: Number of PEMs in the system. The PEM sections are numbered from 0 through *n-1*.

35.5.8 Power Feed Section

The power feed section contains the IPMB address information for the power feeds in the chassis.

The format for this section is:

[PowerFeed1]

IpmbAddress=*IPMB_address_of_power_feed_1*

...

[PowerFeedN]

IpmbAddress=*IPMB_address_of_power_feed_n*

n: Number of power feeds in the system.

35.5.9 Fan section

This section contains information regarding the intelligent fans and the logical device they connect to.

The format for this section is:

```
[Fan]

NumFans=N

Fan0=LogicalDeviceX

...

FanN-1=LogicalDeviceY
```

N: Number of fans in the system

X: Number of logical device connected to Fan0

Y: Number of logical device connected to Fan*N-1*

35.5.10 PEM Section

This section contains information regarding the intelligent power entry modules (PEMs) in the chassis and which logical device they connect to. The format for this section is as follows:

```
[PEM]

NumPEMs=N

PEM0=LogicalDeviceX

...

PEMN-1=LogicalDeviceY
```

N: Number of PEMs in the system

X: Logical device connected to PEM0

Y: Logical device connected to PEM*N-1*

35.6 Installing Configuration Files

The RSM stores chassis configuration files for each chassis in a subdirectory `/etc/cmm/chassis/<chassis_name>`. The chassis name must match the concatenation of the manufacturer's name and product name.

The portion of the directory name for the manufacturer's name must be capitalized.

The `cmm.ini` configuration file needs to be present in the `/etc/cmm/chassis/<chassis_name>` subdirectory.

35.7 Adding Files to RSM

The files created following the instructions in this guide can be added to the RSM in one of two ways. One way is to copy the files manually to the appropriate directory on the RSM using FTP or a comparable method. Another way is to package the files into an OEM.zip file that can be used with the firmware update command. Using this second method, the files in the OEM.zip file are automatically loaded onto the RSM when the `update` command is executed.

35.7.1 Copying Files to RSM Manually

Note: This process needs to be followed on both the active and standby RSMs. You can copy the files to both RSMs in any order, but make sure both RSMs are rebooted after a successful copy.

The configuration files created above can be manually copied to the RSM using FTP or another comparable method. First, create the proper directory under `/etc/cmm/chassis`. The name of this directory must match the manufacturer name field and the product name field in the board area of the FRU. Once the directory has been created, the configuration files can be copied there.

After all the files have been copied, the chassis must be restarted. Upon boot up the RSM will read the appropriate chassis name from the FRU. The RSM then finds the configuration information in the new directory by matching the chassis name in the FRU with the directory name.

35.7.2 Creating OEM.zip File

The new configuration files can be packaged into a `.zip` file with an accompanying `.md5` checksum file. These can then be used in conjunction with the `cmmset -l cmm -d update` command to automatically update the RSMs with the new directory and configuration files.

Follow these steps:

1. Package the new configuration files into a `.zip` file. This file should be named *chassis_name*`.zip`.
   Each file added to the `.zip` file must contain the full path name of the directory into which the file will be extracted on the RSM. For example, if the name of the `chassis` directory is `/etc/cmm/chassis/INTEL_MPCHC0001`, the `.zip` file must include the path `/etc/cmm/chassis/INTEL_MPCHC0001` for each file.
2. Create the accompanying `.md5` file for the checksum with the file name *chassis_name*`.md5`.

On Linux systems you can create the chassis configuration packet (`.zip`, `.md5`) in two steps, assuming all chassis files are in the `INTEL_MPCHC0001` directory:

```
zip -r INTEL_MPCHC0001.zip /etc/cmm/chassis/INTEL_MPCHC0001

md5sum INTEL_MPCHC0001.zip > INTEL_MPCHC0001.md5
```

Once these two files are created, they can be used with the firmware update package and the firmware update command to place new chassis configuration information on the RSM.

35.7.3 Adding Chassis Support using Update Command

To add chassis configuration files with the firmware update process, the same process for a command line firmware update is followed as described in Chapter 32.0, "Updating RSM Software." However, a new `oem` option has been added to the `cmmset -l cmm -d update` command to cater to the processing of a *chassisName*`.zip` file.

The command for doing a firmware update that includes adding chassis configuration files looks like this:

`cmmset -l cmm -d update -v "`*path_and_name_of_CMM_firmware_update_package* `[oem:`*path_and_name_of_chassisName*`.zip_`*file*`]"`

The *path_and_name_of_CMM_firmware_update_package* and *path_and_name_of_chassisName*`.zip_`*file* must include the full pathname for the file.

The `.zip` extension is not included when specifying the path and name of the *chassisName*`.zip` file immediately following the `oem` option.

If the new `oem` option is used with the `cmmset -l cmm -d update` command, the *chassis_name*`.zip` file will be unzipped and verified using the *chassis_name*`.md5` file. If the file is verified, the contents are stored in the `/etc/cmm/chassis/<chassis_name>` directory on the RSM.

After updating the RSMs, you must reboot them so they can read the newly installed configuration information.

35.8 Assumptions and Limitations

This section describes some of the assumptions and limitations that pertain to third party chassis support.

35.8.1 LED Control

This section describes some assumptions and limitations with respect to LEDs.

35.8.1.1 Multicolored LEDs

To control an LED that supports only one color, a single GPIO pin is sufficient. The GPIO pin wired to the LED needs to be driven high to low (or low to high depending on the polarity) to turn the LED on or off. To change the color of a single physical LED that supports two or more colors requires at least two GPIO pins.

The RSM assumes that a single control register is used to drive the output of the GPIO pins that control LEDs that can display more than one color.

35.8.1.2 Health LEDs

Managed FRUs can have one or more health LEDs. The health status of the managed FRU can be indicated by either a single LED that displays multiple colors (one per severity level) or by several LEDs, where each LED is dedicated to a different severity level and each displays a different single color.

In the latter case it is easy to turn on individual LEDs to indicate multiple health events at different severity levels. In the former case the one LED can be illuminated with the color denoting the highest severity level.

35.8.2 Chassis Data Module

This section describes some assumptions and limitations with respect to the Chassis Data Modules (CDMs).

35.8.2.1 CDM LEDs

If the CDMs have LEDs to indicate their health, these LEDs must be controlled by the LED control signals coming from the shelf manager module. See the "A6K-RSM-J Hardware Reference" for more information about these signals.

35.8.3 Sensors

The RSM supports a limited set of sensors on the managed devices. The supported sensors are for temperature, voltage, and fan and entity presence.

The "Filter Run Time" sensor is a special OEM sensor that keeps track of the run time of an air filter. This sensor should be used if a chassis has an air filter tray. If this sensor is added to the chassis SDR, the sensor type value must be 0xC0.

All chassis sensor numbers must lie in the range 1–128. All RSM sensor numbers must lie in the range 129–254. All sensor numbers used in the chassis SDR file must lie in the range 1–254.

35.8.4 Fronted FRU Aliasing

A chassis may house non-intelligent fan trays, PEMs, or air filter trays. An alias for each of these devices must be defined in the `[Alias Output]` section of the `cmm.ini` configuration file.

To ensure alignment with the RSM MIB, the SNMP daemon running on the RSM requires that the following names be used for the aliases in the `cmm.ini` configuration file:

- **Fan Tray**: Define the alias(es) `FanTray`*n* where *n* is the instance ID (*not* the FRU ID) of the managed fan tray. If there are three fan trays, the aliases must be `FanTray1`, `Fantray2`, and `FanTray3`. Because the numeric suffix following `FanTray` denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so both the "F" and the "T" in `FanTray`*n* must be capitalized.
- **Power Entry Module**: Define the aliases `PEM`*n*, where *n* is the instance ID (*not* the FRU ID) of the managed PEM. If there are two PEMs, the aliases must be `PEM1` and `PEM2`. Because the numeric suffix following `PEM` denotes an instance ID, the suffix may or may not match the FRU ID.These aliases are case-sensitive, so `PEM` in `PEM`*n* must be capitalized.
- **Air Filter Tray**: Define the alias `FilterTray`*n* where *n* is the instance ID (*not* the FRU ID) of the managed air filter tray. This alias is case-sensitive, so both the "F" and the "T" in `FilterTray`*n* must be capitalized. There can be no more than one managed filter tray in the chassis.
- **SAP**: Define the aliases `SAP`*n*, where n is the instance ID (not the FRU ID) of the fronted Shelf Alarm Panel. If there are 2 SAP's, the aliases must be `SAP1` and `SAP2`. Because the numeric suffix following SAP denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so all three letter "`S`","`A`"and the "`P`" in `SAP`*n* must be capitalized. If there is only one fronted SAP then *n* should be omitted and the alias should be `SAP`.
- **Shelf FRU**: Define the aliases ShelfFru*n*, where *n* is the instance ID (not the FRU ID) of the fronted Shelf Fru. If there are 2 Shelf Fru's, the aliases must be `ShelfFru1` and `ShelfFru2`. Because the numeric suffix following `ShelfFru` denotes an instance ID, the suffix may or may not match the FRU ID. These aliases are case-sensitive, so both the "`S`" and the "`F`" in `ShelfFru`*n* must be capitalized.

36.0 Agency Information

36.1 North America (FCC Class A)

FCC Verification Notice

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the use will be required to correct the interference at his own expense.

36.2 Canada – Industry Canada (ICES-003 Class A)

CANADA – INDUSTRY CANADA

Cet appareil numérique respecte les limites bruits radioélectriques applicables aux appareils numériques de Classe A prescrites dans la norme sur le matériel brouilleur: “Appareils Numériques”, NMB-003 édictée par le Ministre Canadian des Communications.

(English translation of the notice above) This digital apparatus does not exceed the Class A limits for radio noise emissions from digital apparatus set out in the interference-causing equipment standard entitled “Digital Apparatus,” ICES-003 of the Canadian Department of Communications.

36.3 Safety Instructions

36.3.1 English

CAUTION: This equipment is designed to permit the connection of the earthed conductor of the d.c. supply circuit to the earthing conductor at the equipment. See installation instructions. If this connection is made, all of the following conditions must be met:

-This equipment shall be connected directly to the DC supply system earthing electrode conductor or to a bonding jumper from an earthing terminal bar or bus to which the DC supply system earthing electrode conductor is connected.

-This equipment shall be located in the same immediate area (such as adjacent cabinets) as any other equipment that has a connection between the earthed conductor of the same DC supply circuit and the earthing conductor, and also the point of earthing of the DC system. The DC system shall not be earthed elsewhere.

-The DC supply source shall be located within the same premises as this equipment.

-Switching or disconnecting devices shall be in the earthed circuit conductor between the DC source and the point of connection of the earthing electrode conductor.

36.3.2 French

Cet appareil est conçu pour permettre le raccordement du conducteur relié à la terre du circuit d'alimentation c.c. au conducteur de terre de l'appareil. Cet appareil est conçu pour permettre le raccordement du conducteur relié à la terre du circuit d'alimentation c.c. au conducteur de terre de l'appareil. Pour ce raccordement, toutes les conditions suivantes doivent être respectées:

- Ce matériel doit être raccordé directement au conducteur de la prise de terre du circuit d'alimentation c.c. ou à une tresse de mise à la masse reliée à une barre omnibus de terre laquelle est raccordée à l'électrode de terre du circuit d'alimentation c.c.

- Les appareils dont les conducteurs de terre respectifs sont raccordés au conducteur de terre du même circuit d'alimentation c.c. doivent être installés à proximité les uns des autres (p.ex., dans des armoires adjacentes) et à proximité de la prise de terre du circuit d'alimentation c.c. Le circuit d'alimentation c.c. ne doit comporter aucune autre prise de terre. matériel. - Il ne doit y avoir

– La source d'alimentation du circuit c.c. doit être située dans la même pièce que le aucun dispositif de commutation ou de sectionnement entre le point de raccordement au conducteur de la source d'alimentation c.c. et le point de raccordement à la prise de terre.

36.4 Taiwan Class A Warning Statement

警告使用者 ：
這是甲類的資訊產品， 在居住的環境中使用時，
可能會造成射頻干擾， 在這種情況下，使用者會
被要求採取某些適當的對策。

36.5 Japan VCCI Class A

この装置は、情報処理装置等電波障害自主規制協議会（ＶＣＣＩ）の基準に基づくクラスＡ情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。

36.6 Korean Class A

기종별	사용자 안내문
A급 기기 (업무용 정보통신기기)	이기기는 업무용으로 전자파적합등록을 한 기기이오니 판매자 또는 사용자는 이 점을 주의하시기 바라며 만약 잘못 판매 구입 하였을 때에는 가정용으로 교환하시기 바랍니다.

36.7 Australia, New Zealand

N-232

37.0 Safety Warnings

Caution: Review the following precautions to avoid personal injury and prevent damage to this product or products to which it is connected. To avoid potential hazards, use the product only as specified.

Read all safety information provided in the component product user manuals and understand the precautions associated with safety symbols, written warnings, and cautions before accessing parts or locations within the unit. Save this document for future reference.

AC AND/OR DC POWER SAFETY WARNING: The AC and/or DC Power cord is the unit's main AC and/or DC disconnecting device, and must be easily accessible at all times. Auxiliary AC and/or DC On/Off switches and/or circuit breaker switches are for power control functions only (NOT THE MAIN DISCONNECT).

IMPORTANT: See installation instructions before connecting to the supply.

For AC systems, use only a power cord with a grounded plug and always make connections to a grounded main. Each power cord must be connected to a dedicated branch circuit.

For DC systems, this unit relies on the building's installation for short circuit (over-current) protection. Ensure that a Listed and Certified fuse or circuit breaker no larger than 72VDC, 15A is used on all current carrying conductors. For permanently connected equipment, a readily accessible disconnect shall be incorporated in the building installation wiring. For permanent connections, use copper wire of the gauge specified in the system's user manual.

The enclosure provides a separate Earth ground connection stud. Make the Earth ground connection prior to applying power or peripheral connections and never disconnect the Earth ground while power or peripheral connections exist.

To reduce the risk of electric shock from a telephone or Ethernet* system, connect the unit's main power before making these connections. Disconnect these connections before removing main power from the unit.

RACK MOUNT ENCLOSURE SAFETY: This unit may be intended for stationary rack mounting. Mount in a rack designed to meet the physical strength requirements of NEBS GR-63-CORE and NEBS GR 487. Disconnect all power sources and external connections prior to installing or removing the unit from a rack.

System weight may be minimized prior to mounting by removing all hot-swappable equipment. Mount your system in a way that ensures even loading of the rack. Uneven weight distribution can result in a hazardous condition. Secure all mounting bolts when rack mounting the enclosure.

Warning: Verify power cord and outlet compatibility: Use the appropriate power cords for your power outlet configurations. Visit the following web site for additional information: http:// kropla.com/electric2.htm.

Warning: Avoid electric overload, heat, shock, or fire hazard: Only connect the system to a to a properly rated supply circuit as specified in the product user manual. Do not make connections to terminals outside the range specified for that terminal. See the product user manual for correct connections.

Warning: Avoid electric shock: Do not operate in wet, damp, or condensing conditions. To avoid electric shock or fire hazard, do not operate this product with enclosure covers or panels removed.

Warning: Avoid electric shock: For units with multiple power sources, disconnect all external power connections before servicing.

Warning: Power supplies must be replaced by qualified service personnel only.

Caution: System environmental requirements: Components such as Processor Boards, Ethernet Switches, etc., are designed to operate with external airflow. Components can be destroyed if they are operated without external airflow. External airflow is normally provided by chassis fans when components are installed in compatible chassis. Never restrict the airflow through the unit's fan or vents. Filler panels or air management boards must be installed in unused chassis slots. Environmental specifications for specific products may differ. Refer to product user manuals for airflow requirements and other environmental specifications.

Warning: Device heatsinks may be hot during normal operation: To avoid burns, do not allow anything to touch heatsinks.

Warning: Avoid injury, fire hazard, or explosion: Do not operate this product in an explosive atmosphere.

Caution: Lithium batteries. There is a danger of explosion if a battery is incorrectly replaced or handled. Do not disassemble or recharge the battery. Do not dispose of the battery in fire. When the battery is replaced, the same type (CR2032) or an equivalent type recommended by the manufacturer must be used. Used batteries must be disposed of according to the manufacturer's instructions.

Warning: Avoid injury: This product may contain one or more laser devices that are visually accessible depending on the plug-in modules installed. Products equipped with a laser device must comply with International Electrotechnical Commission (IEC) 60825.

37.1 Mesures de Sécurité

Veuillez suivre les mesures de sécurité suivantes pour éviter tout accident corporel et ne pas endommager ce produit ou tout autre produit lui étant connecté. Pour éviter tout danger, veillez à utiliser le produit conformément aux spécifications mentionnées.

Lisez toutes les informations de sécurité fournies dans les manuels de l'utilisateur des produits composants et veillez à bien comprendre les mesures associées aux symboles de sécurité, aux avertissements écrits et aux mises en garde avant d'accéder à certains éléments ou emplacements de l'unité. Conservez ce document comme outil de référence.

AVERTISSEMENT CONCERNANT LA SÉCURITÉ DE L'ALIMENTATION C.A. ET/OU C.C. : le câble d'alimentation C.A. et/ou C.C. constitue le dispositif de déconnexion principal de l'alimentation électrique de l'unité et doit être facilement accessible à tous moments. Les commutateurs de marche/arrêt C.A. et/ou C.C. et/ou les commutateurs disjoncteurs auxiliaires permettent uniquement de contrôler l'alimentation (ET NON LA DÉCONNEXION PRINCIPALE).

IMPORTANT : reportez-vous aux instructions d'installation avant de connecter le bloc d'alimentation.

Pour les systèmes C.A., utilisez uniquement un câble d'alimentation avec une prise de terre et établissez toujours les connexions à une prise secteur mise à la terre. Chaque câble d'alimentation doit être connecté à un circuit terminal dédié.

Pour les systèmes C.C., la protection de cette unité repose sur les coupe-circuits (surintensité) du bâtiment. Assurez-vous d'utiliser un fusible ou un disjoncteur répertorié et certifié ne dépassant pas 72 VCC et 15 A pour tous les conducteurs de courant. Pour les équipements connectés en permanence, un sectionneur facilement accessible doit être incorporé au câblage du bâtiment. Pour les connexions permanentes, utilisez des câbles en cuivre d'un calibre conforme à celui spécifié dans le manuel de l'utilisateur du système.

Le boîtier fournit un connecteur de mise à la terre séparé. Établissez la connexion à la terre avant de mettre le système sous tension ou de connecter des périphériques. Veillez à ne jamais déconnecter la mise à la terre tant que le système est sous tension ou si des périphériques sont connectés.

Pour réduire le risque d'un choc électrique en provenance d'un téléphone ou d'un système Ethernet*, connectez l'alimentation principale de l'unité avant d'établir ces connexions. De même, déconnectez-les avant de couper l'alimentation principale de l'unité.

SÉCURITÉ DU BOÎTIER POUR UN MONTAGE EN BAIE : cette unité peut être destinée à un montage en baie stationnaire. Le montage en baie doit satisfaire aux exigences sur la résistance physique des normes NEBS GR-63-CORE et NEBS GR 487. Déconnectez toutes les sources d'alimentation et les connexions externes avant d'installer ou de supprimer l'unité d'une baie.

Minimisez la masse du système avant le montage en retirant l'équipement permutable à chaud. Assurez-vous que le système est réparti de manière uniforme sur la baie. Une distribution inégale de la masse du système peut présenter des risques. Fixez tous les boulons lors de l'installation du boîtier dans une baie.

Avertissement : vérifiez que le câble d'alimentation et la prise sont compatibles. Utilisez les câbles d'alimentation correspondant à la configuration de vos prises de courant. Pour de plus amples informations, visitez le site Web suivant : http://kropla.com/electric2.htm.

Avertissement : évitez toute forme de surcharge, chaleur, choc électrique ou incendie. Connectez uniquement le système à un circuit d'alimentation dûment répertorié conformément aux spécifications du manuel de l'utilisateur du produit. N'établissez pas de connexions à des terminaux en dehors des limites spécifiées pour ce terminal. Reportez-vous au manuel de l'utilisateur du produit pour les connections adéquates.

Avertissement : évitez les chocs électriques. N'utilisez pas ce produit dans des endroits humides, mouillés ou provoquant de la condensation. Pour éviter tout risque de choc électrique ou d'incendie, n'utilisez pas ce produit si les couvercles ou les panneaux du boîtier ne sont pas en place.

Avertissement : évitez les chocs électriques. Pour les unités comportant plusieurs sources d'alimentation, déconnectez toutes les sources d'alimentation externes avant de procéder aux réparations.

Avertissement : les blocs d'alimentation doivent être remplacés exclusivement par des techniciens d'entretien qualifiés.

Attention : exigences environnementales du système : les composants tels que les cartes de processeurs, les commutateurs Ethernet, etc., sont conçus pour fonctionner avec un flux d'air externe. Les composants peuvent être détruits s'ils fonctionnent dans d'autres conditions. Le flux d'air externe est généralement produit par les ventilateurs des châssis lorsque les composants sont installés dans des châssis compatibles. Veillez à ne jamais obstruer le flux d'air alimentant le ventilateur ou les conduits de l'unité. Des boucliers ou des panneaux de gestion de l'air doivent être installés dans les connecteurs inutilisés du châssis. Les spécifications environnementales peuvent varier d'un produit à un autre. Veuillez-vous reporter au manuel de l'utilisateur pour déterminer les exigences en matière de flux d'air et d'autres spécifications environnementales.

Avertissement : les dissipateurs de chaleur de l'appareil peuvent être chauds lors d'un fonctionnement normal. Pour éviter tout risque de brûlure, veillez à ce que rien n'entre en contact avec les dissipateurs de chaleur.

Avertissement : évitez les blessures, les incendies ou les explosions. N'utilisez pas ce produit dans une atmosphère présentant des risques d'explosion.

Attention : les batteries au lithium. Celles-ci peuvent exploser si elles sont incorrectement remplacées ou manipulées. Veillez à ne pas désassembler ni à recharger la batterie. Veillez à ne pas jeter la batterie au feu. Lors du remplacement de la batterie, utilisez le même type de batterie (CR2032) ou un type équivalent recommandé par le fabricant. Les batteries usagées doivent être mises au rebut conformément aux instructions du fabricant.

Avertissement : évitez les blessures. Ce produit peut contenir un ou plusieurs périphériques laser visuellement accessibles en fonction des modules plug-in installés. Les produits équipés d'un périphérique laser doivent être conformes à la norme IEC (International Electrotechnical Commission) 60825.

37.2 Sicherheitshinweise

Lesen Sie bitte die folgenden Sicherheitshinweise, um Verletzungen und Beschädigungen dieses Produkts oder der angeschlossenen Produkte zu verhindern. Verwenden Sie das Produkt nur gemäß den Anweisungen, um mögliche Gefahren zu vermeiden.

Lesen Sie alle Sicherheitsinformationen in den Benutzerhandbüchern der zu dem Produkt gehörenden Komponenten und machen Sie sich mit den Hinweisen zu den Sicherheitssymbolen, schriftlichen Warnungen und Vorsichtsmaßnahmen vertraut, ehe Sie Teile oder Stellen des Geräts anfassen. Bewahren Sie dieses Dokument gut auf, um später darin nachlesen zu können.

SICHERHEITSWARNUNG FÜR WECHSELSTROM UND/ODER GLEICHSTROM: Die Stromversorgung des Gerätes wird über das Wechselstrom- und/oder Gleichstromkabel unterbrochen und muss daher jederzeit leicht zugänglich sein. Zusätzliche Ein-/Aus-Schalter für Wechselstrom und/oder Gleichstrom und/oder Leistungsschalter dienen lediglich der Steuerung der Stromversorgung (NICHT ABER DER UNTERBRECHUNG DER STROMVERSORGUNG).

WICHTIG: Lesen Sie vor dem Anschließen der Stromversorgung die Installationsanweisungen!

Wechselstromsysteme: Verwenden Sie nur ein Stromkabel mit geerdetem Stecker und verbinden Sie dieses immer nur mit einer geerdeten Steckdose. Jedes Stromkabel muss an einen eigenen Stromkreis angeschlossen werden.

Gleichstromsysteme: Dieses Gerät basiert auf dem im Gebäude installierten Schutz vor Kurzschlüssen (Netzüberlastung). Stellen Sie sicher, dass für alle stromführenden Leiter eine zertifizierte Sicherung oder ein Leistungsschalter mit nicht mehr als 72V Gleichstrom, 15A verwendet wird. Für Geräte, die ständig angeschlossen sind, sollte in der Gebäudeverkabelung ein leicht zugänglicher Trennschalter installiert werden. Für eine permanente Verbindung verwenden Sie Kupferdraht der im Benutzerhandbuch des Systems angegebenen Stärke.

Das Gehäuse verfügt über einen eigenen Erdungs-Verbindungsbolzen. Stellen Sie die Erdungsverbindung her, ehe Sie das Stromkabel oder Peripheriegeräte anschließen, und trennen Sie die Erdungsverbindung niemals, so lange Strom- und Peripherieverbindungen angeschlossen sind.

Um die Gefahr eines durch ein Telefon oder Ethernet*-System bedingten elektrischen Schlags zu verringern, schließen Sie das Stromkabel des Geräts an, ehe Sie diese Verbindungen einrichten. Trennen Sie diese Verbindungen, ehe Sie die Hauptstromversorgung des Geräts unterbrechen.

SICHERHEITSHINWEISE BEI GESTELLMONTAGE: Dieses Gerät kann stationär in einem Gestell angebracht werden. Das Gestell muss den Anforderungen an eine physische Stärke laut NEBS GR-63-CORE und NEBS GR 487 entsprechen. Trennen Sie vor der Installation oder dem Abbau des Geräts in einem Gestell alle Strom- und externen Verbindungen.

Das Gewicht des Systems kann vor dem Einbau verringert werden, indem man alle während des Betriebs austauschbaren Elemente entfernt. Achten Sie darauf, das System so aufzustellen, dass das Gestell gleichmäßig belastet wird. Eine ungleiche Verteilung des Gewichts kann gefährlich werden. Befestigen Sie alle Sicherungsbolzen, wenn Sie das Gehäuse in einem Gestell montieren.

Warnung: Überprüfen Sie, ob Stromkabel und Steckdose kompatibel sind: Verwenden Sie die Ihrer Stromkonfiguration entsprechenden Stromkabel. Weitere Informationen finden Sie auf folgender Website: http://kropla.com/electric2.htm.

Warnung: Vermeiden Sie elektrische Überlastung, Hitze, elektrischen Schlag oder Feuergefahr: Schließen Sie das System nur an einen den Spezifikationen des Produkt-Benutzerhandbuchs entsprechenden Stromkreis an. Stellen Sie keine Verbindung zu Terminals her, die nicht den jeweiligen Spezifikationen entsprechen. Für die korrekten Verbindungen siehe das Benutzerhandbuch des Produkts.

Warnung: Vermeiden Sie einen elektrischen Schlag: Unterlassen Sie den Betrieb in nassen, feuchten oder kondensierenden Betriebsumgebungen. Um die Gefahr eines elektrischen Schlags oder eines Feuers zu vermeiden, betreiben Sie dieses Produkt nicht ohne Gehäuse oder Abdeckungen.

Warnung: Vermeiden Sie einen elektrischen Schlag: Trennen Sie bei Geräten mit mehreren Stromquellen vor der Wartung alle externen Stromverbindungen.

Warnung: Netzteile dürfen nur von qualifizierten Servicemitarbeitern ausgewechselt werden.

Vorsicht: Anforderungen an die Systemumgebung: Komponenten wie Prozessor-Boards, Ethernet-Schalter usw. sind auf den Betrieb mit externer Luftzufuhr ausgelegt. Diese Komponenten können bei Betrieb ohne externe Luftzufuhr beschädigt werden. Wenn die Komponenten in einem kompatiblen Gehäuse installiert sind, wird Luft von außen normalerweise durch Gehäuselüfter zugeführt. Blockieren Sie niemals die Luftzufuhr der Gerätelüfter oder -ventilatoren. In ungenutzten Gehäusesteckplätzen müssen Füllelemente oder Luftsteuerungseinheiten eingesetzt werden. Die Betriebsbedingungen können zwischen den verschiedenen Produkten variieren. Für die Anforderungen an die Belüftung und andere Betriebsbedingungen siehe die Benutzerhandbücher der jeweiligen Produkte.

Warnung: Die Kühlkörper des Geräts können sich während des normalen Betriebs erhitzen: Um Verbrennungen zu vermeiden, sollte jeder Kontakt mit den Kühlkörpern vermieden werden.

Warnung: Vermeiden Sie Verletzungen, Feuergefahr oder Explosionen: Unterlassen Sie den Betrieb dieses Produkts in einer explosionsgefährdeten Betriebsumgebung.

Vorsicht: Lithiumbatterien. Bei unsachgemäßem Austausch oder Umgang mit Batterien besteht Explosionsgefahr. Zerlegen Sie die Batterie nicht und laden Sie diese nicht wieder auf. Entsorgen Sie die Batterie nicht durch Verbrennen. Beim Auswechseln der Batterie muss dasselbe oder ein der Händlerempfehlung gleichwertiges Modell verwendet werden (CR2032). Gebrauchte Batterien müssen entsprechend den Anweisungen des Herstellers entsorgt werden.

Warnung: Vermeiden Sie Verletzungen: Dieses Produkt kann ein oder mehrere Lasergeräte enthalten, die abhängig von den installierten Plug-In-Modulen optisch zugänglich sind. Mit einem Lasergerät ausgestattete Produkte müssen der International Electrotechnical Commission (IEC) 60825 entsprechen.

37.3 Norme di Sicurezza

Leggere le norme seguenti per prevenire lesioni personali ed evitare di danneggiare questo prodotto o altri a cui è collegato. Per evitare qualsiasi pericolo potenziale, usare il prodotto unicamente come indicato.

Leggere tutte le informazioni sulla sicurezza fornite nella guida per l'utente relativa al componente e comprendere le norme associate ai simboli di pericolo, agli avvisi scritti e alle precauzioni da adottare prima di accedere a componenti o aree dell'unità. Custodire il presente documento per usi futuri.

AVVISO DI SICUREZZA RELATIVO ALL'ALIMENTAZIONE IN C.A. E/O C.C. Il cavo di alimentazione in c.a. e/o c.c. rappresenta il dispositivo principale per interrompere l'alimentazione in c.a. e/o c.c. dell'unità e deve sempre essere facilmente accessibile. Gli interruttori di accensione/ spegnimento ausiliari per l'alimentazione in c.a. e/o c.c. hanno l'unico scopo di controllare l'alimentazione (NON INTERROMPONO L'ALIMENTAZIONE PRINCIPALE).

IMPORTANTE: prima di collegare l'unità alla fonte di alimentazione, leggere le istruzioni di installazione.

Per i sistemi CA, usare solo un cavo di alimentazione con una spina provvista di una messa a terra e collegarsi sempre a prese provviste di una messa a terra. Ogni cavo di alimentazione deve essere collegato ad un circuito derivato dedicato.

Per i sistemi CC, la presente unità può usufruire dell'eventuale installazione integrata nell'edificio per la protezione contro i cortocircuiti (sovratensione). Assicurarsi della presenza di un fusibile o di un circuito derivato non superiore a 72 V c.c., 15 A, certificato e conforme alla normativa in vigore, in tutti i conduttori portanti. Per gli apparecchi collegati in modo permanente, è necessario inserire nel circuito dell'edificio un interruttore ad accesso immediato. Per i collegamenti permanenti, usare il filo di rame del diametro specificato nella guida per l'utente relativa al sistema.

Il materiale fornito comprende un perno per il collegamento della messa a terra. Assicurare il collegamento della messa a terra prima di alimentare l'unità o prima di collegarla alle periferiche e non scollegare mai la messa a terra quando l'unità è alimentata o collegata a periferiche.

Per ridurre il rischio di scariche elettriche da parte della linea telefonica o dalla rete Ethernet*, collegare l'unità all'alimentazione principale prima di effettuare tale collegamento. Rimuovere i collegamenti prima di togliere l'alimentazione principale all'unità.

NORME DI SICUREZZA PER LE UNITÀ MONTATE IN UN RACK. Questa unità può essere alloggiata in modo permanente in un rack. Il montaggio in rack deve essere conforme ai requisiti di resistenza fisica delle norme NEBS GR-63-CORE e NEBS GR 487.Prima di installare o rimuovere l'unità da un rack, rimuovere tutte le fonti di alimentazione e i collegamenti esterni.

Prima di effettuare il montaggio, è possibile ridurre il peso complessivo del sistema togliendo tutte le apparecchiature sostituibili a caldo. Montare il sistema in modo da garantire una distribuzione uniforme del peso nel rack. Una distribuzione irregolare del peso può essere pericolosa. Avvitare fino in fondo tutti i bulloni durante l'installazione dell'unità in un rack.

Avvertenza: verificare il cavo di alimentazione e la compatibilità con la presa di corrente. Usare i cavi di alimentazione compatibili con il tipo di presa di corrente. Per ulteriori informazioni, visitare il sito Web all'indirizzo seguente: http://kropla.com/electric2.htm.

Avvertenza: evitare sovraccarichi elettrici, calore diretto, scosse e possibili cause di incendio. Collegare il sistema solo ad una rete elettrica la cui tensione nominale corrisponda al valore indicato nella guida per l'utente. Non collegarlo a fonti di alimentazione con valori di tensione esterne a quanto specificato per il sistema. Per ulteriori informazioni sul corretto collegamento, consultare la guida per l'utente del prodotto.

Avvertenza: evitare le scosse elettriche. Non usare l'apparecchio in ambienti umidi o in presenza di condensa. Per evitare scosse elettriche o possibili cause di incendio, non adoperare il prodotto senza le custodie o i pannelli appositi.

Avvertenza: evitare le scosse elettriche. Prima di intervenire su unità con più fonti di alimentazione, rimuovere tutti i collegamenti all'alimentazione esterna.

Avvertenza: far sostituire i componenti di alimentazione solo da personale tecnico qualificato.

Attenzione: rispettare i requisiti ambientali del sistema. I componenti come le schede di processore, i commutatori Ethernet, ecc., sono progettati per funzionare in presenza di un flusso di aria proveniente dall'esterno, in assenza del quale rischiano di danneggiarsi irrimediabilmente. In genere, il flusso di aria esterno viene generato da appositi ventilatori installati contemporaneamente ai componenti nello chassis compatibile. Non ostacolare mai il flusso di aria convogliato dal ventilatore e dai condotti dell'unità. I pannelli di copertura o le schede per il controllo dell'aria devono essere installati negli alloggiamenti vuoti dello chassis. I requisiti ambientali possono variare a seconda del prodotto. Per ulteriori informazioni sui requisiti del flusso di aria e sugli altri requisiti ambientali, consultare la guida per l'utente del prodotto.

Avvertenza: i dissipatori di calore possono scaldarsi durante il funzionamento normale. Per evitare bruciature o danni, evitare il contatto del dissipatore di calore con qualsiasi altro elemento.

Avvertenza: evitare lesioni, possibili cause di incendio o di esplosione. Non usare il prodotto in un'atmosfera in cui sussiste il rischio di esplosione.

Attenzione: le batterie al litio. La sostituzione o l'uso non corretto della batteria comporta un rischio di esplosione. Non smontare né ricaricare la batteria. Non gettare la batteria nel fuoco. Per la sostituzione, usare il tipo di batteria identico (CR2032) o equivalente consigliato dal costruttore. Le batterie usate devono essere smaltite rispettando le istruzioni del costruttore.

Avvertenza: evitare le lesioni. Questo prodotto può contenere uno o più dispositivi laser accessibili alla vista, a seconda dei moduli installati. I prodotti provvisti di un dispositivo laser devono essere conformi alla norma 60825 della Commissione elettrotecnica internazionale (IEC).

37.4 Instrucciones de Seguridad

Examine las instrucciones sobre condiciones de seguridad que siguen para evitar cualquier tipo de daños personales, así como para evitar perjudicar el producto o productos a los que esté conectado. Para evitar riesgos potenciales, utilice el producto únicamente en la forma especificada.

Lea toda la información relativa a seguridad que se incluye en los manuales de usuario de los distintos componentes y procure familiarizarse con los distintos símbolos de seguridad, advertencias escritas y normas de precaución antes de manipular las distintas piezas o secciones de la unidad. Guarde este documento para consultarlo en el futuro.

AVISO DE SEGURIDAD SOBRE LA ALIMENTACIÓN DE CA O CC El cable de alimentación de CA o CC constituye el dispositivo principal de desconexión de la alimentación de CA o CC, y debe permanecer accesible en todo momento. Los interruptores auxiliares de encendido y apagado de CA o CC y los disyuntores sólo tienen una función de control de la alimentacion (Y NO LA DE DESCONEXIÓN PRINCIPAL).

IMPORTANTE: Consulte las instrucciones de instalación antes de conectar la unidad a la alimentación.

En el caso de sistemas de CA, utilice sólo cables de alimentación con enchufe con toma de tierra, y realice siempre conexiones a una toma con toma de tierra. Cada uno de los cables de alimentación deberá estar conectado a una derivación dedicada.

En el caso de sistemas de CC, la unidad dependerá de la instalación existente en el edificio para la protección frente a cortocircuitos (sobreintensidades). Asegúrese de que todos los conductores que transporten corriente empleen un fusible o disyuntor homologado y certificado con una capacidad que no supere los 72V de CC ni 15A. En el caso de los equipos que vayan a permanecer conectados de manera constante, en la instalación eléctrica del edificio deberá estar incluida una desconexión de fácil acceso. Para conexiones permanentes, emplee cable de cobre del calibre especificado en el manual de usuario del sistema.

El chasis incluye aparte una clavija de conexión a tierra. Realice la conexión a tierra antes de suministrar corriente o realizar cualquier tipo de conexión de periféricos; no desconecte nunca la toma de tierra mientras la corriente esté presente o existan conexiones con periféricos.

Para reducir los riesgos de descargas eléctricas a través de un teléfono o un sistema de Ethernet*, conecte la alimentación principal de la unidad antes de realizar este tipo de conexiones. Desconecte estas conexiones antes de desconectar la alimentación principal de la unidad.

PROCEDIMIENTOS DE SEGURIDAD PARA EL CHASIS DE MONTAJE EN BASTIDOR: Esta unidad puede estar preparada para su montaje en un bastidor estático. Un montaje de este tipo deberá realizarse en un bastidor que cumpla con los requisitos de robustez de las normas NEBS GR-63-CORE y NEBS GR 487. Desconecte cualquier tipo de alimentación y conexiones externas antes de instalar la unidad en un bastidor o desmontarla.

Puede desmontar todos los equipos de intercambio en caliente para reducir el peso del sistema antes del montaje en bastidor. Asegúrese de montar el sistema de forma que el peso quede distribuido uniformemente en el bastidor. Una distribución irregular del peso podría generar riesgos. Asegúrese de fijar todos los tornillos de montaje en el bastidor.

Advertencia: Compatibilidad del cable y la toma: Utilice los cables adecuados para la configuración de tomas de corriente con que cuente. Si necesita más información, visite el sitio web siguiente: http://kropla.com/electric2.htm.

Advertencia: Evite sobrecargas eléctricas, calor y riesgos de descarga eléctrica o incendio: Conecte el sistema sólo a un circuito de alimentación que tenga el régimen apropiado, según lo especificado en el manual de usuario del producto. No realice conexiones con terminales cuya capacidad no se ajuste al régimen especificado para ellos. Consulte el manual de usuario del producto para que las conexiones que realice sean las correctas.

Advertencia: Evite descargas eléctricas: No haga funcionar el sistema en condiciones de humedad, mojado o si se produce condensación de la humedad. Para evitar descargas eléctricas o posibles incendios, no permita que el aparato funcione con sus tapas o paneles del chasis desmontados.

Advertencia: Evite descargas eléctricas: En el caso de unidades que cuenten con varias fuentes de alimentación, desconecte las conexiones con alimentación externa antes de proceder a realizar labores de mantenimiento.

Advertencia: La sustitución de fuentes de alimentación sólo debe ser realizada por personal de mantemiento cualificado.

Precaución: Requisitos de entorno para el sistema: Los componentes del tipo de placas de procesador, conmutadores de Ethernet, etc., están concebidos para funcionar en condiciones que permitan el paso de aire. Los componentes pueden averiarse si funcionan sin que circule el aire en su entorno. La circulación del aire suele estar facilitada por los ventiladores incorporados en el armazón cuando los componentes están instalados en armazones compatibles. Nunca interrumpa el paso del aire por los ventiladores or los respiraderos. Los paneles de relleno y las placas para el control de la circulación del aire deben instalarse en ranuras del chasis que no estén destinadas a ningún otro uso. Las características técnicas relativas al entorno pueden variar entre productos. Consulte los manuales de usuario del producto si necesita conocer sus necesidades en términos de circulación de aire u otras características técnicas.

Advertencia: En condiciones de funcionamiento normales, los disipadores de calor pueden recalentarse. Evite que ningún elemento entre en contacto con los disipadores para evitar quemaduras.

Advertencia: Riesgos de daños, incendio o explosión: No permita que el aparato funcione en una atmósfera que presente riesgos de explosión.

Precaución: Las baterías de litio. Si las baterías no se manipulan o cambian correctamente, exite riesgo de explosión. No desmonte ni recargue la batería. Nunca tire las baterías al fuego. Al cambiar la batería, es preciso utilizar el mismo tipo (CR2032) o un tipo equivalente que haya sido recomendado por el fabricante. Las baterías utilizadas deben desecharse según las instrucciones del fabricante.

Advertencia: Daños personales: Este producto puede contener uno o varios dispositivos láser, que estarán a la vista dependiendo de los módulos enchufables que se hayan instalado. Los productos provistos de un dispositivo láser deben ajustarse a la norma 60825 de la International Electrotechnical Commission (IEC).

37.5 Chinese Safety Warning

系统信息

请阅读以下警告信息，以避免人身伤害并防止损坏本产品或与之相连的产品。为避免潜在的危险，请仅按规定使用产品。

在接近设备中的部件或元件之前，请阅读在组件产品用户手册中载明的所有安全信息并了解与安全标志、书面警告及注意事项有关的预防措施。请保存本文档以备将来参考。

交流电和（或）直流电源安全警告：交流电和（或）直流电源线是设备的主要交流电和（或）直流电连接装置，任何时候均必须方便取用。辅助交流电和（或）直流电开关和（或）断路器开关仅用于控制电源（非主要断电装置）。

重要：在连接到电源前，请先参阅安装说明。

对于交流电系统，请仅使用带接地插头的电源线，并始终与接地插座保持连接。每根电源线必须连接到专用分支电路。

对于直流电系统，本设备要求建筑物安装短路（过电流）保护装置。请确保在所有载流导线中使用不大于 72VDC、15A，经过认证和鉴定的保险丝或断路器。对于永久连接设备，应在建筑物布线中安装便于断开的装置。对于永久连线，应使用系统用户手册中指定的标准铜线。

机壳配备了单独的接地接线柱。请先接好接地线后再通电或连接其它线路，在通电或接通周边电线时，切勿断开接地线。

为降低电话或 Ethernet* 系统电击的危险，请在连接设备主电源后，再连接其它线路；在从设备上卸下主电源前，请先断开这些线路。

机架装配或机壳的安全性：本设备采用固定机架装配。机架装配的设计符合 NEBS GR-63-CORE 和 NEBS GR 487 的机械强度要求。在机架上安装或拆卸设备前，务必断开所有电源和外部连线。

卸下所有热交换设备可最大限度地减少系统重量。小心装配系统以确保机架载重均衡，载重分配不均衡可能导致危险情况发生。当机架上安装机壳时，请上紧所有装配螺栓。

警告：请检查电源线与电源插座是否兼容。请使用与电源插座配置对应的电源线。有关详情，请访问下面的网址：http://kropla.com/electric2.htm。

警告：避免电力过载、过热、电击或火灾：仅将系统连接到产品用户手册中指定的适当额定供电电路。请勿连接到超出接线端规定范围的接线端。有关正确连接，请参阅产品用户手册。

警告：避免电击：请勿在潮湿或冷凝条件下运行。为避免电击或火灾，请勿在卸下机壳或面板的情况下使用本产品。

警告：避免电击：对于具有多个电源的设备，请在维修前断开所有外部电源连接。

警告：更换电源只能由合格的维修人员进行。

注意：系统环境要求：处理器插件板、以太网开关等组件要求在外部空气流通的环境下工作。在没有外部空气流通的情况下操作，可能会损坏组件。当组件安装在机架机箱中时，外部气流通常是由机箱风扇提供的。切勿阻碍气流通过的设备风扇或通风口。在不使用的机箱插槽中，必须安装填充板或空气控制板。环境规范因特定产品而异。有关气流要求和其它环境规范，请参阅产品用户手册。

警告：设备散热片在正常运行期间可能发热：为避免燃烧，请勿让任何物体接触散热片。

警告：避免人身伤害、火灾或爆炸：切勿在可能引起爆炸的环境中使用本产品。

注意：锂电池不是可现场更换的部件。如果更换或处理不当，可能会引起爆炸。切勿拆掉电池或对电池充电，切勿让电池靠近火源。更换电池时，必须使用制造商建议的相同类型或同等类型的电池。旧电池必须按照制造商的指示加以处理，并将电池退回给英特尔修理。

警告：避免辐射：本产品可能包含一个或多个激光装置。是否可见取决于所安装的插件模块。装有激光装置的产品必须遵循国际电工委员会 (IEC) 60825 号规定。

Appendix A Sensor Numbers

A.1 Shelf Sensors

Shelf sensors are available on shelf manager IPMB address 20h. They are seen as targets on CLI location "chassis" (except for event-only sensors). The numbers are valid for the Radisys MPCHC0001 chassis. Numbers for other chassis types may vary.

Table 71. Shelf Sensors (sheet 1 of 2)

Number	Name (ID String)	Sensor Type	References
0Ah	FilterTrayTemp1	01h	Table 77, “Generic Sensors from IPMI v1.5 Table 36-2” on page 216
0Bh	FilterTrayTemp2	01h	Table 77, “Generic Sensors from IPMI v1.5 Table 36-2” on page 216
0Ch	Filter Run Time	C0h	Table 159, “Filter Run Time Sensor” on page 270
43h	Filter Tray HS	F0h	Table 117, “PICMG Hot Swap Sensor” on page 245
4Dh	Filter Tray	25h	Table 112, “Entity Presence Sensor from IPMI 1.5 Spec, Table 36-3” on page 242
4Eh	Air Filter	25h	Table 112, “Entity Presence Sensor from IPMI 1.5 Spec, Table 36-3” on page 242
5Fh	CDM 2	25h	Table 112, “Entity Presence Sensor from IPMI 1.5 Spec, Table 36-3” on page 242
60h	CDM 1	25h	Table 112, “Entity Presence Sensor from IPMI 1.5 Spec, Table 36-3” on page 242
0x8B	IPMB-0 Snsr 1	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x8C	IPMB-0 Snsr 2	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x8D	IPMB-0 Snsr 3	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x8E	IPMB-0 Snsr 4	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x8F	IPMB-0 Snsr 5	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x90	IPMB-0 Snsr 6	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x91	IPMB-0 Snsr 7	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x92	IPMB-0 Snsr 8	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x93	IPMB-0 Snsr 9	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x94	IPMB-0 Snsr 10	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x95	IPMB-0 Snsr 11	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x96	IPMB-0 Snsr 12	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x97	IPMB-0 Snsr 13	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x98	IPMB-0 Snsr 14	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x99	IPMB-0 Snsr 15	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9A	IPMB-0 Snsr 16	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9B	IPMB-0 Snsr 17	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9C	IPMB-0 Snsr 18	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9D	IPMB-0 Snsr 19	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9E	IPMB-0 Snsr 20	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0x9F	IPMB-0 Snsr 21	F1h	Table 120, “PICMG IPMB-0 Link Sensor” on page 247
0xA0	Log Usage	10h	Table 92, “Event Logging Disabled Sensor from IPMI 1.5 Spec, Table 36-3” on page 230 (event only)
0xA1	NonCompliant FRU	CBh	Table 158, “Non Compliant FRU Sensor” on page 269 (event only)
0xA2	Power Allocation	CCh	Table 147, “Power Allocation Sensor” on page 264 (event only)
0xA3	Cooling Policy	CAh	Table 149, “Cooling Policy Sensor” on page 265
0xA4	Temp Condition	CEh	Table 150, “Temperature Condition Sensor” on page 265

Table 71. Shelf Sensors (sheet 2 of 2)

Number	Name (ID String)	Sensor Type	References
0xA5	ReEnum Status	CFh	Table 151, “Re-enumeration Sensor” on page 266 (event only)
0xA6	PowerRestoreFail	D6h	Table 164, “Power Restoration Failure” on page 273 (event only)
0xE0	Power Budget 1	CDh	Table 148, “Power Budget Sensor” on page 265
0xE1	Power Budget 2	CDh	Table 148, “Power Budget Sensor” on page 265
0xE2	Power Budget 3	CDh	Table 148, “Power Budget Sensor” on page 265
0xE3	Power Budget 4	CDh	Table 148, “Power Budget Sensor” on page 265

A.2 RSM Sensors

The physical IPMC monitors various on-board sensors to determine the health status of the board. The IPMC takes appropriate actions in the event of a hardware or software failure, such as lighting LEDs and generating events.

The RSM implements the following types of sensors.

- **Discrete** — A discrete sensor can have up to 16 bit-mapped states, with one state as true.
- **Digital** — A digital sensor has two possible states, only one of which can be active at any given time. For example, a digital sensor monitoring the power may have a state detecting whether the power is good or the power is not good.
- **OEM** — An OEM sensor has its states defined by the manufacturer. The reading types of these sensors are sometimes defined as “sensor-specific.”
- **Threshold** — A threshold sensor has a range of 256 values, which represent measurements on the RSM and its FRUs. Temperature, voltage, current, and fan speed sensors are examples of threshold sensors.

The possible thresholds are listed in Table 72.

Table 72. Threshold types

Threshold Type	Description
UNR	Upper non-recoverable thresholds generate a critical alarm on the high side.
UC	Upper critical thresholds generate a major alarm on the high side.
UNC	Upper non-critical thresholds generate a minor alarm on the high side.
LNC	Lower non-critical thresholds generate a minor alarm on the low side.
LC	Lower critical thresholds generate a major alarm on the low side
LNR	Lower non-recoverable thresholds typically generate a critical alarm on the low side

A.2.1 RSM Sensors - Physical IPMC

The tables in this section describe the physical IPMC managed sensors supported by the RSM. The thresholds are based on the voltage and temperature requirements of the devices present. The column labeled “Normal Reading” shows the normal sensor reading in a byte format. These sensors appear as targets on CLI location "cmm" (except for event-only sensors).

Table 73. RSM sensors available on physical address, LUN 00 (sheet 1 of 2)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
0	FRU 0 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides blade FRU 0 M state hot swap information as defined in the ATCA specification.
1	Version Change	IPMI Version Change	Sensor specific discrete	N/A	Yes	N/A	N/A	Reports firmware version changes as defined in the IPMI v2.0 specification.
2	ATCA IPMB-0	ATCA IPMB-0 Sensor	Sensor specific discrete	0x0088	Yes	N/A	N/A	Reports IPMB-0 operational status as defined in the ATCA specification.
3	IPMC Reset	OEM IPMC Reset	Digital discrete	N/A	Yes	N/A	N/A	Generates an event when the IPMC is reset.
4	LMP Reset	OEM Payload Reset	Sensor specific discrete	N/A	Yes	N/A	N/A	Generates an event when the LMP is reset.
5	CFD Watchdog	OEM CFD Watchdog	Sensor specific discrete	N/A	Yes	N/A	N/A	Event-only SDR type. Sensor will not be displayed in `listargets` report.
6	BMC Watchdog	Watchdog 2	Sensor specific discrete	N/A	Yes	N/A	N/A	Event-only SDR type. Sensor will not be displayed in `listargets` report.
7	Ejector Closed	Slot/ Connector	Digital discrete	N/A	Yes	N/A	N/A	Reports the status of the hot swap ejector latch.
8	-48V Absent A	Power Supply	Digital discrete	0x0001	Yes	N/A	N/A	Reports the status of -48V input A..
9	-48V Absent B	Power Supply	Digital discrete	0x0001	Yes	N/A	N/A	Reports the status of -48V input B.
10	-48V Fuse Fault	Power Supply	Digital discrete	0x0001	Yes	N/A	N/A	Reports the status of the -48V fuses.
11	ShMC-X BusA Rdy	Slot/ Connector	Digital discrete	0x0002	Yes	N/A	N/A	Ready status for the ShMC cross connect IPMB-0 bus A.
12	ShMC-X BusB Rdy	Slot/ Connector	Digital discrete	0x0002	Yes	N/A	N/A	Ready status for the ShMC cross connect IPMB-0 bus B.

Table 73. RSM sensors available on physical address, LUN 00 (sheet 2 of 2)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
13	+12V	Voltage	Threshold	12.0	Yes	Minor, Major, Critical	0.15V	See Table 9, "RSM Sensor Thresholds" on page 31 for default threshold values. Event generation is disabled for the +3.0V Battery sensor when the RSM is used in an NECCH0001 chassis.
14	+3.6V I2C A	Voltage	Threshold	3.60	Yes	Minor, Major, Critical	0.04V	
15	+3.6V I2C B	Voltage	Threshold	3.60	Yes	Minor, Major, Critical	0.04V	
16	+3.3V	Voltage	Threshold	3.30	Yes	Minor, Major, Critical	0.04V	
17	+3.0V Battery	Voltage	Threshold	3.00	Yes (See Notes)	Minor, Major, Critical	0.04V	
18	+2.5V	Voltage	Threshold	2.50	Yes	Minor, Major, Critical	0.03V	
19	+1.8V	Voltage	Threshold	1.80	Yes	Minor, Major, Critical	0.02V	
20	+1.2V	Voltage	Threshold	1.20	Yes	Minor, Major, Critical	0.02V	
21	+1.05V CPU Core	Voltage	Threshold	1.05	Yes	Minor, Major, Critical	0.02V	
22	+0.9V	Voltage	Threshold	0.90	Yes	Minor, Major, Critical	0.01V	
23	CPU Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	
24	ADM1026 Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	
25	IPMC Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	

See Table 9, "RSM Sensor Thresholds" on page 31 for additional information about the managed sensors for the physical IPMC.

Table 74. RSM event only sensors

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Notes
40	Sys FW Progress	System Firmware Progress	OEM 0x70	N/A	Events are generated by the LMP processor as it progresses through its boot process.
41	IPMC HA State	OEM 0xD0	Sensor specific discrete	N/A	An event is generated when the IPMC changes its redundant state. Event byte 2 is new state and event byte 3 is old state: 0x10 = active 0x03 = standby
42	IPMC Failover	OEM 0xD1	Sensor specific discrete	N/A	An event is generated when the IPMC begins failover and another when failover processing is complete. Event byte 2 indicates failover state: 0 = failover start 1 = failover complete Event byte 3 indicates the failover reason for debug purposes: 1 = communication lost with active peer IPMC 2 = peer IPMC is not active 4 = Set Redundant Status command received 6 = both IPMCs are active

Table 75. RSM sensors available on physical address, LUN 02

Number	Name (ID String)	Sensor Type	References
60	RT Diagnostics	C2h	Table 152, “RT Diagnostics Sensor” on page 267
61	Reboot Reason	C4h	Table 154, “Reboot Reason Sensor” on page 268
62	PMS Health	C7h	Table 141, “PMS Health Sensor” on page 261
63	HA trap connect	C5h	Table 124, “HA Trap Connect Sensor” on page 248
64	NTP Status	C6h	Table 157, “NTP Status Sensor” on page 269
65	DataSync Status	DEh	Table 133, “DataSync Status Sensor” on page 254
66	HA state	C9h	Table 127, “HA State Sensor” on page 250
67	CMM Status	D9h	Table 162, “CMM Status Sensor” on page 272
68	HA redundancy	C8h	Table 135, “HA Redundancy Sensor” on page 256
69	HA OOS Request	DCh	Table 125, “HA Out of Service Request Sensor” on page 249
70	HA INS Request	DDh	Table 126, “HA In Service Request Sensor” on page 249
Event-only sensors			
71	PMS Fault	DAh	Table 139, “PMS Fault Sensor” on page 259 (event only)
72	PMS Info	DBh	Table 140, “PMS Info Sensor” on page 260 (event only)
73	Security	E0h	Table 155, “Security Sensor” on page 268 (event only)
74	HA Peer Lost	D5h	Table 163, “HA Peer Lost Sensor” on page 272 (event only)
75	HA Health Score	D3h	Table 134, “HA Health Score Sensor” on page 255 (event only)
76	HA control	D2h	Table 136, “HA Control Sensor” on page 257 (event only)
77	Local Upgrade	DFh	Table 142, “Local Upgrade Sensor” on page 262 (event only)

A.2.2 RSM Sensors - Virtual IPMC

The virtual IPMC and its sensors are only represented by the active shelf manager. Depending on the shelf type, certain sensors may not be present.

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 1 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
Virtual FRU 0 sensors								
0	FRU 0 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 0 blade M state hot swap information as defined in the ATCA specification.
1	FRU 1 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 1 shelf FRU info M state hot swap information as defined in the ATCA specification.
2	FRU 2 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 2 shelf FRU info M state hot swap information as defined in the ATCA specification.
3	FRU 3 Hot Swap	PICMG ATCA Hot Swap	Digital discrete	N/A	Yes	N/A	N/A	Provides FRU 3 SAP M state hot swap information as defined in the ATCA specification.
4	FRU 4 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 4 Fan Tray 1 M state hot swap information as defined in the ATCA specification.
5	FRU 5 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 5 Fan Tray 2 M state hot swap information as defined in the ATCA specification.
6	FRU 6 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 6 Fan Tray 3 M state hot swap information as defined in the ATCA specification.
7	FRU 7 Hot Swap	PICMG ATCA Hot Swap	Digital discrete	N/A	Yes	N/A	N/A	Provides FRU 7 PEM A M state hot swap information as defined in the ATCA specification.
8	FRU 8 Hot Swap	PICMG ATCA Hot Swap	Sensor specific discrete	N/A	Yes	N/A	N/A	Provides FRU 8 PEM B M state hot swap information as defined in the ATCA specification.
9	Ejector Closed	Slot/ Connector	Digital discrete	0x01	No	N/A	N/A	Reports the status of the hot swap latch for FRU 0.
10	CDM 1	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for CDM 1 FRU 1.
11	CDM 2	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for CDM 2 FRU 2.
12	SAP	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for SAP FRU 3.
13	Fan Tray 1	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for fan tray 1 FRU 4
14	Fan Tray 2	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for fan tray 2 FRU 5
15	Fan Tray 3	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for fan tray 3 FRU 6
16	PEM A	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for PEM A FRU 7
17	PEM B	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for PEM B FRU 8
18	Air Filter	Entity Presence	Sensor specific	0x01	Yes	Major	N/A	Presence indicator for the air filter
19	+24V Fan Fault	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of +24V to fans

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 2 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
20	Slot 1 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 1 IPMB-0 bus A
21	Slot 1 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 1 IPMB-0 bus B
22	Slot 2 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 2 IPMB-0 bus A
23	Slot 2 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 2 IPMB-0 bus B
24	Slot 3 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 3 IPMB-0 bus A
25	Slot 3 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 3 IPMB-0 bus B
26	Slot 4 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 4 IPMB-0 bus A
27	Slot 4 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 4 IPMB-0 bus B
28	Slot 5 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 5 IPMB-0 bus A
29	Slot 5 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 5 IPMB-0 bus B
30	Slot 6 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 6 IPMB-0 bus A
31	Slot 6 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 6 IPMB-0 bus B
32	Slot 7 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 7 IPMB-0 bus A
33	Slot 7 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 7 IPMB-0 bus B
34	Slot 8 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 8 IPMB-0 bus A
35	Slot 8 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 8 IPMB-0 bus B
36	Slot 9 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 9 IPMB-0 bus A
37	Slot 9 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 9 IPMB-0 bus B
38	Slot 10 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 10 IPMB-0 bus A
39	Slot 10 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 10 IPMB-0 bus B
40	Slot 11 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 11 IPMB-0 bus A
41	Slot 11 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 11 IPMB-0 bus B
42	Slot 12 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 12 IPMB-0 bus A
43	Slot 12 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 12 IPMB-0 bus B
44	Slot 13 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 13 IPMB-0 bus A
45	Slot 13 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 13 IPMB-0 bus B
46	Slot 14 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 14 IPMB-0 bus A

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 3 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
47	Slot 14 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 14 IPMB-0 bus B
48	Slot 15 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 15 IPMB-0 bus A
49	Slot 15 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 15 IPMB-0 bus B
50	Slot 16 BusA Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 16 IPMB-0 bus A
51	Slot 16 BusB Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for Slot 16 IPMB-0 bus B
52	Chassis Bus 0 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 0
53	Chassis Bus 1 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 1
54	Chassis Bus 2 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 2
55	Chassis Bus 3 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 3
56	Chassis Bus 4 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 4
57	Chassis Bus 5 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 5
58	Chassis Bus 6 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 6
59	Chassis Bus 7 Rdy	Slot/ Connector	Digital discrete	0x02	Yes	N/A	N/A	Ready status for chassis I2C interface 7
RSM sensor SDRs								
100	Temp Condition	The IPMC lists sensor SDRs on behalf of the RSM software (LUN 2), which requires them to be present in order to function. They are listed here since they are present in the IPMI firmware and must fit into its sensor table numbering.						
101	Cooling Policy							
102	Power Budget 1							
103	Power Budget 2							
104	Power Budget 3							
105	Power Budget 4							
106	Power Budget 5							
107	Power Budget 6							
108	Power Budget 7							
109	Power Budget 8							
RSM event only sensor SDRs								
110	Log usage	The IPMC lists event only sensor SDRs on behalf of the RSM software (LUN 2), which requires them to be present in order to function. They are listed here since they are present in the IPMI firmware and must fit into its sensor table numbering.						
111	NonCompliantFRU							
112	PowerRestoreFail							
113	ReEnumStatus							
114	Power Allocation							

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 4 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
Virtual FRU 1 sensors								
120	FRU 1 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for CDM1, always closed
121	CDM 1 Health	CDM Health	OEM	0x02	Yes	N/A	N/A	Sensor will not scan and log events if CDM 1 is not present. Events are logged if a `read/write fru` command fails when it is sent to the IPMC. An event is also logged if the CDM 1 contents differ from the write data in the `Write FRU data` command.
Virtual FRU 2 sensors								
122	FRU 2 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for CDM2, always closed
123	CDM 2 Health	CDM Health	OEM	0x02	Yes	N/A	N/A	Sensor will not scan and log events if CDM 2 is not present. Events are logged if a `read/write fru` command fails when it is sent to the IPMC. An event is also logged if the CDM 2 contents differ from the write data in the `Write FRU data` command.
Virtual FRU 3 sensors								
124	FRU 3 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for SAP
125	Telco Alrm Input	PICMG Telco Input	Sensor Specific Discrete	0x00	Yes	N/A	N/A	Telco alarm input sensor as defined in the ATCA specification
126	SAP Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 | LC: -5 | LNC: 0 | UNC: 65 | UC: 72 | UNR: 80
Virtual FRU 4 sensors								
127	FRU 4 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for fan tray 1
128	-48A Bus Flt 1	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A input bus
129	-48A Fuse Flt 1	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A after fuse on fan tray
130	-48B Bus Flt 1	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B input bus
131	-48B Fuse Flt 1	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B after fuse on fan tray
132	+24V Fault 1	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of +24V input
133	Left Output Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 | LC: -5 | LNC: 0 | UNC: 65 | UC: 72 | UNR: 80
134	Fan 1 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 5 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
135	Fan 2 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
94	Fan 3 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
Virtual FRU 5 sensors								
136	FRU 5 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for fan tray 1
137	-48A Bus Flt 2	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A input bus
138	-48A Fuse Flt 2	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A after fuse on fan tray
139	-48B Bus Flt 2	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B input bus
140	-48B Fuse Flt 2	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B after fuse on fan tray
141	+24V Fault 2	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of +24V input
142	Cntr Output Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 | LC: -5 | LNC: 0 | UNC: 65 | UC: 72 | UNR: 80
143	Fan 4 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
144	Fan 5 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
104	Fan 6 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
Virtual FRU 6 sensors								
145	FRU 6 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for fan tray 1
146	-48A Bus Flt 3	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A input bus
147	-48A Fuse Flt 3	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V A after fuse on fan tray
148	-48B Bus Flt 3	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B input bus
149	-48B Fuse Flt 3	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of -48V B after fuse on fan tray

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 6 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
150	+24V Fault 3	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of +24V input
151	Rght Output Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 LC: -5 LNC: 0 UNC: 65 UC: 72 UNR: 80
152	Fan 7 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
153	Fan 8 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
114	Fan 9 Speed	Fan	Threshold	N/A	Yes	Minor, Major, Critical	100RPM	This sensor measures temperature in RPM Thresholds are read-only and variable inside the firmware depending on the fan speed setting
Virtual FRU 7 sensors								
154	FRU 7 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for PEM A
155	PEM A In 1 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 1 of the PEM
156	PEM A Fuse 1 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 1 fuse of the PEM
157	PEM A In 2 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 2 of the PEM
158	PEM A Fuse 2 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 2 fuse of the PEM
159	PEM A In 3 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 3 of the PEM
160	PEM A Fuse 3 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 3 fuse of the PEM
161	PEM A In 4 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 4 of the PEM
162	PEM A Fuse 4 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 4 fuse of the PEM
163	PEM A Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 LC: -5 LNC: 0 UNC: 65 UC: 72 UNR: 80
Virtual FRU 8 sensors								
164	FRU 8 Latch Clsd	Slot/ Connector	Digital discrete	0x02	No	N/A	N/A	Hot swap latch status for PEM B
165	PEM B In 1 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 1 of the PEM
166	PEM B Fuse 1 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 1 fuse of the PEM

Table 76. RSM sensors available on virtual address, LUN 02 (sheet 7 of 7)

Sensor Number	Name (ID String)	Sensor Type	Reading Type	Normal Reading	Event Generation	Alarm Level	Hysteresis	Notes
167	PEM B In 2 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 2 of the PEM
168	PEM B Fuse 2 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 2 fuse of the PEM
169	PEM B In 3 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 3 of the PEM
170	PEM B Fuse 3 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 3 fuse of the PEM
171	PEM B In 4 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 4 of the PEM
172	PEM B Fuse 4 Flt	Power Supply	Digital discrete	0x01	Yes	N/A	N/A	Reports the status of input 4 fuse of the PEM
173	PEM B Temp	Temp	Threshold	25	Yes	Minor, Major, Critical	2°C	This sensor measures temperature in °C Default Threshold LNR: -10 | LC: -5 | LNC: 0 | UNC: 65 | UC: 72 | UNR: 80

A.2.3 Device Sensor Data Record (SDR) Repository

The ATCA specification requires the IPMC to maintain a Sensor Data Record (SDR) repository for the sensors that the board manages. This SDR repository provides the access methods for the shelf manager to gather sensor information.

The IPMC firmware implements the SDR repository within program memory. Threshold value settings modified by IPMI commands are not preserved over power cycles of the IPMC.

Appendix B IPMI Generic Sensor Events

B.1 Introduction

This appendix documents the sensors listed in Table 36-2 of the IPMI Specification Version 1.5 Revision1.1 that are implemented in the A6K-RSM-J shelf manager module firmware.

B.2 Explanation of Abbreviations and Symbols

This section explains the column heading abbreviations and special symbols used in the tables in this appendix.

- RTC means Reading Type Code
- ERC means Event Reading Class
- OF means Generic Offset
- SH means System Health contribution
- (A) means Assertion
- (D) means Deassertion
- Dash (–) means “not applicable”.

B.3 Event Severity and Contribution to System Health

The severity (OK, Minor, Major, Critical) of the event listed in the table, whether for assertion (A) or deassertion (D), is the default used by the RSM firmware when the sensor does not provide its own severity setting.

If the SH (System Health) column indicates “No” for an event code, it means that the severity of the event does not contribute to system health by default.

Table 77. Generic Sensors from IPMI v1.5 Table 36-2 (sheet 1 of 5)

RTC	ERC	OF	Event Code[a]	Event Description	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
01h	Threshold	00h	0010	Lower Non-critical - going low (A)	Lower non-critical going low: Assertion	Minor	—	Yes
			001C	Lower Non-critical - going low (D)	Lower non-critical going low: Deassertion	—	OK	Yes
		01h	0011	Lower Non-critical - going high (A)	Lower non-critical going high: Assertion	Minor	—	Yes
			001D	Lower Non-critical - going high (D)	Lower non-critical going high: Deassertion	—	OK	Yes
		02h	0012	Lower Critical - going low (A)	Lower critical going low: Assertion	Major	—	Yes
			001E	Lower Critical - going low (D)	Lower critical going low: Deassertion	—	OK	Yes
		03h	0013	Lower Critical - going high (A)	Lower critical going high: Assertion	Major	—	Yes
			001F	Lower Critical - going high (D)	Lower critical going high: Deassertion	—	OK	Yes
		04h	0014	Lower Non-recoverable - going low (A)	Lower non-recoverable going low: Assertion	Critical	—	Yes
			0020	Lower Non-recoverable - going low (D)	Lower non-recoverable going low: Deassertion	—	OK	Yes
		05h	0015	Lower Non-recoverable - going high (A)	Lower non-recoverable going high: Assertion	Critical	—	Yes
			0021	Lower Non-recoverable - going high (D)	Lower non-recoverable going high: Deassertion	—	OK	Yes
		06h	0016	Upper Non-critical - going low (A)	Upper non-critical going low: Assertion	Minor	—	Yes
			0022	Upper Non-critical - going low (D)	Upper non-critical going low: Deassertion	—	OK	Yes
		07h	0017	Upper Non-critical - going high (A)	Upper non-critical going high: Assertion	Minor	—	Yes
			0023	Upper Non-critical - going high (D)	Upper non-critical going high: Deassertion	—	OK	Yes
		08h	0018	Upper Critical - going low (A)	Upper critical going low: Assertion	Major	—	Yes
			0024	Upper Critical - going low (D)	Upper critical going low: Deassertion	—	OK	Yes
01h	Threshold	09h	0019	Upper Critical - going high (A)	Upper critical going high: Assertion	Major	—	Yes
			0025	Upper Critical - going high (D)	Upper critical going high: Deassertion	—	OK	Yes
		0Ah	001A	Upper Non-recoverable - going low (A)	Upper non-recoverable going low: Assertion	Critical	—	Yes
			0026	Upper Non-recoverable - going low (D)	Upper non-recoverable going low: Deassertion	—	OK	Yes
		0Bh	001B	Upper Non-recoverable - going high (A)	Upper non-recoverable going high: Assertion	Critical	—	Yes
			0027	Upper Non-recoverable - going high (D)	Upper non-recoverable going high: Deassertion	—	OK	Yes

Table 77. Generic Sensors from IPMI v1.5 Table 36-2 (sheet 2 of 5)

RTC	ERC	OF	Event Code[a]	Event Description	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
02h	Discrete	00h	1020	Transition to Idle	Transition to Idle: Assertion	OK	—	No
			1021		Transition to Idle: Deassertion	—	OK	No
		01h	1022	Transition to Active	Transition to Active: Assertion	OK	—	No
			1023		Transition to Active: Deassertion	—	OK	No
		02h	1024	Transition to Busy	Transition to Busy: Assertion	OK	—	No
			1025		Transition to Busy: Deassertion	—	OK	No
03h	Digital Discrete	00h	1030	State Deasserted (A)	State Deassertion: Assertion	OK	—	No
			1031	State Deasserted (D)	State Deassertion: Deassertion	—	OK	No
		01h	1032	State Asserted (A)	State Assertion: Assertion	OK	—	No
			1033	State Asserted (D)	State Assertion: Deassertion	—	OK	No
04h	Digital Discrete	00h	1040	Predictive Failure deasserted	Predictive Failure deasserted: [Assertion|Deassertion]	OK	OK	Yes
		01h	1041	Predictive Failure asserted	Predictive Failure asserted: [Assertion|Deassertion]	Minor	OK	Yes
05h	Digital Discrete	00h	1050	Limit Not Exceeded	Limit Not Exceeded: [Assertion|Deassertion]	OK	OK	Yes
		01h	1051	Limit Exceeded	Limit Exceeded: [Assertion|Deassertion]	Minor	OK	Yes
06h	Digital Discrete	00h	1060	Performance Met	Performance Met: [Assertion|Deassertion]	OK	OK	No
		01h	1061	Performance Lags	Performance Lags: [Assertion|Deassertion]	OK	OK	No

Table 77. Generic Sensors from IPMI v1.5 Table 36-2 (sheet 3 of 5)

RTC	ERC	OF	Event Code[a]	Event Description	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
07h	Discrete	00h	1070	transition to OK	transition to OK: [Assertion|Deassertion]	OK	OK	Yes
		01h	1071	transition to Non-Critical from OK	transition to Non-Critical from OK: [Assertion|Deassertion]	Minor	OK	Yes
		02h	1072	transition to Critical from less severe	transition to Critical from less severe: [Assertion|Deassertion]	Major	OK	Yes
		03h	1073	transition to Non-recoverable from less severe	transition to Non-recoverable from less severe: [Assertion|Deassertion]	Critical	OK	Yes
		04h	1074	transition to Non-Critical from more severe	transition to Non-Critical from more severe: [Assertion|Deassertion]	Minor	OK	Yes
		05h	1075	transition to Critical from Non-recoverable	transition to Critical from Non-recoverable: [Assertion|Deassertion]	Major	OK	Yes
		06h	1076	transition to Non-recoverable	transition to Non-recoverable: [Assertion|Deassertion]	Critical	OK	Yes
		07h	1077	Monitor	Monitor: [Assertion|Deassertion]	OK	OK	Yes
		08h	1078	Informational	Informational: [Assertion|Deassertion]	OK	OK	Yes
08h	Digital Discrete	00h	0040	Device Removed / Device Absent (A)	Device Removed: Assertion	Major	—	Yes
			0041	Device Removed / Device Absent (D)	Device Removed: Deassertion	—	OK	Yes
		01h	0042	Device Inserted / Device Present (A)	Device Inserted: Assertion	OK	—	Yes
			0043	Device Inserted / Device Present (D)	Device Inserted: Deassertion	—	Major	Yes
09h	Digital Discrete	00h	1090	Device Disabled	Device Disabled: [Assertion|Deassertion]	OK	OK	No
		01h	1092	Device Enabled	Device Enabled: [Assertion|Deassertion]	OK	OK	No

Table 77. Generic Sensors from IPMI v1.5 Table 36-2 (sheet 4 of 5)

RTC	ERC	OF	Event Code[a]	Event Description	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
0Ah	Discrete	00h	10A0	transition to Running	transition to Running: [Assertion|Deassertion]	OK	OK	Yes
		01h	10A1	transition to Test	transition to Test: [Assertion|Deassertion]	OK	OK	Yes
		02h	10A2	transition to Power Off	transition to Power Off: [Assertion|Deassertion]	OK	OK	Yes
		03h	10A3	transition to On Line	transition to On Line: [Assertion|Deassertion]	OK	OK	Yes
		04h	10A4	transition to Off Line	transition to Off Line: [Assertion|Deassertion]	OK	OK	Yes
		05h	10A5	transition to Off Duty	transition to Off Duty: [Assertion|Deassertion]	OK	OK	Yes
		06h	10A6	transition to Degraded	transition to Degraded: [Assertion|Deassertion]	OK	OK	Yes
		07h	10A7	transition to Power Save	transition to Power Save: [Assertion|Deassertion]	OK	OK	Yes
		08h	10A8	Install Error	Install Error: [Assertion|Deassertion]	Minor	OK	Yes
0Bh	Discrete	00h	10B0	Fully Redundant	Fully Redundant: [Assertion|Deassertion]	OK	OK	Yes
		01h	10B1	Redundancy Lost	Redundancy Lost: [Assertion|Deassertion]	Major	OK	Yes
		02h	10B2	Redundancy Degraded	Redundancy Degraded: [Assertion|Deassertion]	Minor	OK	Yes
		03h	10B3	Non-redundant: Redundancy Lost	Non-redundant: Redundancy Lost: [Assertion|Deassertion]	Major	OK	Yes
		04h	10B4	Non-redundant: Unit regained minimum resources	Non-redundant: Unit regained minimum resources: [Assertion|Deassertion]	Major	OK	Yes
		05h	10B5	Non-redundant: Insufficient Resources	Non-redundant: Insufficient Resources: [Assertion|Deassertion]	Critical	OK	Yes
		06h	10B6	Redundancy Degraded from Fully Redundant	Redundancy Degraded from Fully Redundant: [Assertion|Deassertion]	Minor	OK	Yes
		07h	10B7	Redundancy Degraded from Non-redundant	Redundancy Degraded from Non-redundant: [Assertion|Deassertion]	Minor	OK	Yes

Table 77. Generic Sensors from IPMI v1.5 Table 36-2 (sheet 5 of 5)

RTC	ERC	OF	Event Code[a]	Event Description	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
0Ch	Discrete	00h	10C0	ACPI Device D0 Power State	ACPI Device D0 Power State: [Assertion|Deassertion]	OK	OK	No
		01h	10C1	ACPI Device D1 Power State	ACPI Device D1 Power State: [Assertion|Deassertion]	OK	OK	No
		02h	10C2	ACPI Device D2 Power State	ACPI Device D2 Power State: [Assertion|Deassertion]	OK	OK	No
		03h	10C3	ACPI Device D3 Power State	ACPI Device D3 Power State: [Assertion|Deassertion]	OK	OK	No

a. Event Codes are in hexadecimal.

Appendix C IPMI Typed Sensor Events

C.1 Introduction

This appendix documents the sensors listed in Table 36-3 of the IPMI Specification version 1.5 Revision 1.1.

If there is more than one assertion event for a given offset, the deassertion event for an offset deasserts only the corresponding assertion; assertions for other offsets remain in effect.

Note: The events listed in the table apply only if the Event Reading Code is 6Fh in accordance with the IPMI Specification.

C.2 Explanation of Abbreviations and Symbols

This section explains the column heading abbreviations and special symbols used in the tables in this appendix.

- STC means Sensor Type Code
- OF means Sensor-specific Offset
- ED2 means Event Data 2
- ED3 means Event Data 3
- EC means Event code (in hexadecimal notation)
- SH means System Health contribution
- (A) means Assertion
- (D) means Deassertion
- Dash (–) means “not applicable”.

** means see Appendix B, “IPMI Generic Sensor Events” to determine the value for this cell in the table.

C.3 IPMI Typed Sensor Tables

This section contains the tables for the various sensors that the shelf manager module recognizes from Table 36-3 of the IPMI Specification.

Table 78. Temperature Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Temperature	01h				**	Temperature	**	**	**	Yes

Table 79. Voltage Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Voltage	02h				**	Voltage	**	**	**	Yes

Table 80. Current Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Current	03h				**	Current	**	**	**	Yes

Table 81. Fan Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Fan	04h				**	Fan	**	**	**	Yes

Table 82. Physical Security Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Physical Security (Chassis Intrusion)	05h	00h			0280	General Chassis Intrusion	General Chassis Intrusion: [Assertion|Deassertion]	Major	OK	Yes
		01h			0281	Drive Bay intrusion	Drive Bay intrusion: [Assertion|Deassertion]	Major	OK	Yes
		02h			0282	I/O Card area intrusion	I/O Card area intrusion: [Assertion|Deassertion]	Major	OK	Yes
		03h			0283	Processor area intrusion	Processor area intrusion: [Assertion|Deassertion]	Major	OK	Yes
		04h			0284	LAN Leash Lost (ED2 identifies NIC[b])	LAN Leash Lost[, LAN %ED2[c]]: [Assertion|Deassertion]	Major	OK	Yes
			00h			1st NIC	LAN Leash Lost, LAN 0: [Assertion|Deassertion]	Major	OK	Yes
			*nn*h			*n*th NIC	LAN Leash Lost, LAN %ED2: [Assertion|Deassertion]	Major	OK	Yes
			FFh			NIC not specified	LAN Leash Lost: [Assertion|Deassertion]	Major	OK	Yes
		05h			0285	Unauthorized dock/ undock	Unauthorized dock/ undock: [Assertion|Deassertion]	Major	OK	Yes
		06h			0286	FAN area intrusion	FAN area intrusion: [Assertion|Deassertion]	Major	OK	Yes

a. Event Codes are in hexadecimal.
b. Network Interface Card
c. Value of ED2

Table 83. Platform Security Violation Attempt Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Platform Security Violation Attempt	06h	00h			0510	Secure Mode (Front Panel Lockout) Violation attempt	Secure Mode Violation attempt: [Assertion|Deassertion]	Minor	OK	Yes
		01h			0511	Pre-boot Password Violation - user pwd	Pre-boot Password Violation - user pwd: [Assertion|Deassertion]	Minor	OK	Yes
		02h			0512	Pre-boot Password Violation attempt - setup pwd	Pre-boot Password Violation - setup pwd: [Assertion|Deassertion]	Minor	OK	Yes
		03h			0513	Pre-boot Password Violation - network boot pwd	Pre-boot Password Violation - network boot pwd: [Assertion|Deassertion]	Minor	OK	Yes
		04h			0514	Other pre-boot Password Violation	Other pre-boot Password Violation: [Assertion|Deassertion]	Minor	OK	Yes
		05h			0515	Out-of-band Access Password Violation	Out-of-band Access Password Violation: [Assertion|Deassertion]	Minor	OK	Yes

a. Event Codes are in hexadecimal.

Table 84. Processor Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 1 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Processor	07h	00h			0220	IERR (A)	Processor IERR detected: Assertion	Critical	-	Yes
						IERR (D)	Processor IERR detected: Deassertion	-	OK	Yes
		01h			0221	Thermal Trip (A)	Thermal trip detected: Assertion	Critical	-	Yes
						Thermal Trip (D)	Thermal trip detected: Deassertion	-	OK	Yes
		02h			0222	FRB1/BIST failure (A)	FRB1/BIST failure: Assertion	Critical	-	Yes
						FRB1/BIST failure (D)	FRB1/BIST failure: Deassertion	-	OK	Yes
		03h			0223	FRB2/Hang in POST failure (A)	FRB2/Hang in POST failure: Assertion	Critical	-	Yes
						FRB2/Hang in POST failure (D)	FRB2/Hang in POST failure: Deassertion	-	OK	Yes
		04h			0224	FRB3/Process Startup/ Init failure (CPU no start) (A)	FRB3/Processor Startup/Initialization failure: Assertion	Critical	-	Yes
						FRB3/Process Startup/ Init failure (CPU no start) (D)	FRB3/Processor Startup/Initialization failure: Deassertion	-	OK	Yes
		05h			0225	Configuration Error (A)	Configuration Error detected: Assertion	Critical	-	Yes
						Configuration Error (D)	Configuration Error detected: Deassertion	-	OK	Yes
		06h			0226	SM BIOS 'Uncorrectable CPU-complex Error (A)	SM BIOS - Uncorrectable CPU-complex error: Assertion	Critical	-	Yes
						SM BIOS 'Uncorrectable CPU-complex Error (D)	SM BIOS - Uncorrectable CPU-complex error: Deassertion	-	OK	Yes
		07h			0227	Process Presence detected (A)	Processor Presence detected: Assertion	OK	-	Yes
						Process Presence detected (D)	Processor Presence detected: Deassertion	-	OK	Yes
		08h			0228	Processor disabled (A)	Processor disabled: Assertion	OK	-	Yes
						Processor disabled (D)	Processor disabled: Deassertion	-	OK	Yes

Table 84. Processor Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 2 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Processor	07h	09h			0229	Terminator Presence Detected (A)	Terminator presence detected: Assertion	OK	-	Yes
						Terminator Presence Detected (D)	Terminator presence detected: Deassertion	-	OK	Yes
		0Ah			0230	Processor Automatically Throttled (A)	Processor automatically throttled: Assertion	OK	-	Yes
						Processor Automatically Throttled (D)	Processor automatically throttled: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 85. Power Supply Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Power Supply	08h	00h			0035	Presence detected (A)	Power Supply detected: Assertion	OK	-	Yes
						Presence detected (D)	Power Supply detected: Deassertion	-	Major	Yes
		01h			0031	Power Supply Failure detected (A)	Power Supply Failure detected: Assertion	Critical	-	Yes
						Power Supply Failure detected (D)	Power Supply Failure detected: Deassertion	-	OK	Yes
		02h			0032	Predictive Failure (A)	Power Supply Degraded: Assertion	Minor	-	Yes
						Predictive Failure (D)	Power Supply Degraded: Deassertion	-	OK	Yes
		03h			0033	Power Supply input lost (AC/DC) (A)	Power Supply feed lost: Assertion	Major	-	Yes
						Power Supply input lost (AC/DC) (D)	Power Supply feed lost: Deassertion	-	OK	Yes
		04h			0034	Power Supply input lost or out-of-range (A)	Power Supply feed lost or out of range: Assertion	Critical	-	Yes
						Power Supply input lost or out-of-range (D)	Power Supply feed lost or out of range: Deassertion	-	OK	Yes
		05h			0037	Power Supply input out-of-range, but present (A)	Power Supply feed out of range but present: Assertion	Minor		Yes
						Power Supply input out-of-range, but present (D)	Power Supply feed out of range but present: Deassertion		OK	Yes
		06h			0038	Configuration Error [b]	Power Supply configuration error%ED3[c]: [Assertion|Deassertion]	Minor	OK	Yes
				00h		Vendor Mismatch	- vendor mismatch			
				01h		Revision mismatch	- revision mismatch			
				02h		Processor mission	- processor missing			

a. Event Codes are in hexadecimal.
b. Bits [3:0] of ED3 indicate type of configuration error.
c. Type of configuration error indicated in ED3.

Table 86. Power Unit Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Power Unit	09h	00h			0490	Power Off / Power Down (A)	Power Off: Assertion	OK		Yes
						Power Off / Power Down (D)	Power Off: Deassertion		OK	Yes
		01h			0491	Power Cycle (A)	Power Cycle: Assertion	OK		Yes
						Power Cycle (D)	Power Cycle: Deassertion		OK	Yes
		02h			0492	240VA Power Down (A)	240VA Power Down: Assertion	Major		Yes
						240VA Power Down (D)	240VA Power Down: Deassertion		OK	Yes
		03h			0493	Interlock Power Down (A)	Interlock Power Down: Assertion	Major		Yes
						Interlock Power Down (D)	Interlock Power Down: Deassertion		OK	Yes
		04h			0494	AC lost (A)	AC Lost: Assertion	Major		Yes
						AC lost (D)	AC Lost: Deassertion		OK	Yes
		05h			0495	Soft Power Control Failure (A)	Soft Power Control Failure: Assertion	Major		Yes
						Soft Power Control Failure (D)	Soft Power Control Failure: Deassertion		OK	Yes
		06h			0496	Power Unit Failure detected (A)	Power Unit Failure Detected: Assertion	Major		Yes
						Power Unit Failure detected (D)	Power Unit Failure Detected: Deassertion		OK	Yes
		07h			0497	Predictive Failure (A)	Predictive Failure: Assertion	Major		Yes
						Predictive Failure (D)	Predictive Failure: Deassertion		OK	Yes

a. Event Codes are in hexadecimal.

Table 87. Cooling Device Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Cooling Device	0Ah	-				-	-	-	-	-

Table 88. Other Units-based Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Other Units-based Sensor[a]	0Bh	-				-				

a. Units are supplied in the Sensor Data Record.

Table 89. Memory Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 1 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Memory	0Ch	00h			0240	Correctable ECC/ other corr mem error (A)	Correctable ECC/Other correctable memory error%ED3[b]: Assertion	OK	-	Yes
						Correctable ECC/ other corr mem error (D)	Correctable ECC/Other correctable memory error%ED3: Deassertion	-	OK	Yes
		01h			0241	Uncorrectable ECC (A)	Uncorrectable ECC/ Other uncorrectable memory error%ED3: Assertion	Critical	-	Yes
						Uncorrectable ECC (D)	Uncorrectable ECC/ Other uncorrectable memory error%ED3: Deassertion	-	OK	Yes
		02h			0242	Parity (A)	Parity error detected%ED3: Assertion	Critical	-	Yes
						Parity (D)	Parity error detected%ED3: Deassertion	-	OK	Yes
		03h			0243	Memory Scrub Failed (A)	Memory scrub failed (stuck bit)%ED3: Assertion	Critical	-	Yes
						Memory Scrub Failed (D)	Memory scrub failed (stuck bit)%ED3: Deassertion	-	OK	Yes
		04h			0244	Memory Device Disabled (A)	Memory device disabled%ED3: Assertion	Major	-	Yes
						Memory Device Disabled (D)	Memory device disabled%ED3: Deassertion	-	OK	Yes
		05h			0245	Correctable ECC/ other corr mem err log limit reached (A)	Correctable ECC/Other correctable memory error logging limit reached%ED3: Assertion	Minor	-	Yes
						Correctable ECC/ other corr mem err log limit reached (D)	Correctable ECC/Other correctable memory error logging limit reached%ED3: Deassertion	-	OK	Yes
		06h			0246	Presence detected (A)	Memory presence detected%ED3: Assertion	OK	-	Yes
						Presence detected (D)	Memory presence detected%ED3: Deassertion	-	Major	Yes
		07h			0247	Configuration Error (A)	Memory configuration error%ED3: Assertion	Minor	-	Yes
						Configuration Error (D)	Memory configuration error%ED3: Deassertion	-	OK	Yes

Table 89. Memory Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 2 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Memory	0Ch	08h			0248	Spare Memory (A)	Spare memory%ED3: Assertion	OK	-	Yes
						Spare Memory (D)	Spare memory%ED3: Deassertion	-	OK	Yes
				XX[c]			- Module/Device ID 0x%02X	-	-	-

a. Event Codes are in hexadecimal.
b. All references to %ED3 in the table refer to the value of ED3.
c. Module/Device ID (in hexadecimal)

Table 90. Drive Slot (Bay) Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Drive Slot (Bay)	0Dh	-				-	-			

Table 91. POST Memory Resize Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
POST Memory Resize	0Eh	-				-	-			

Table 92. Event Logging Disabled Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Event Logging Disabled	10h	00h	XXh[b]		0540	Correctable Memory Error Logging Disabled	Correctable Memory Error Logging Disabled, DIMM 0x02%X: [Assertion|Deassertion]	OK	OK	No
		01h			0541	Event 'Type' Logging Disabled	Event 'Type' Logging Disabled	OK	OK	No
			XXh			Event/Reading Type Code				
				XXh		ED3 - [7:6] - reserved. ED3 - [5] - If set, logging has been disabled for all events of the given type ED3 - [4] - Set is assertion event, clear is deassertion event ED3 - [3:0] - Event Offset	0 = Offset %x [assertions|deassertions] 1 = All [assertion|deassertion] events, Event Type 0x%02X: [Assertion|Deassertion]			
				XXh						
				XXh						
				XXh						
		02h			0542	Log Area Reset / Cleared	Log Area Reset/Cleared: [Assertion|Deassertion]	OK	OK	Yes
		03h			0543	All Event Logging Disabled	All Event Logging Disabled: [Assertion|Deassertion]	OK	OK	Yes
		04h			0544	SEL Full	SEL Full: [Assertion|Deassertion]	OK	OK	Yes
		05h			0545	SEL Almost Full	SEL Almost Full %ED3[c]%: [Assertion|Deassertion]	OK	OK	Yes

a. Event Codes are in hexadecimal.
b. ED2 indicates memory module / device id.
c. ED3 indicates percentage of SEL that is filled.

Table 93. System Event Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 1 of 2)

Sensor Type	STC	OF[a]	ED2	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Event	12h	00h			0290	System Reconfigured (A)	System Reconfigured: Assertion	OK	-	Yes
						System Reconfigured (D)	System Reconfigured: Deassertion	-	OK	Yes
		01h			0291	OEM System Boot Event (A)	OEM System boot event: Assertion	OK	-	Yes
						OEM System Boot Event (D)	OEM System boot event: Deassertion	-	OK	Yes
		02h			0292	Undetermined System HW Failure (A)	Undetermined system hardware failure: Assertion	Major	-	Yes
						Undetermined System HW Failure (D)	Undetermined system hardware failure: Deassertion	-	OK	Yes
		03h			0293	Entry added to Aux Log	-	OK	OK	Yes
						ED2 - 7:4 Log Entry Action	The string represented by the high nibble of ED2 is %ED2[7:4][c]			
			00xxh		xxx0	Entry added	%ED2[4:0] entry added: [Assertion|Deassertion]			
			01xxh		xxx1	Entry added because non-IPMI event	%ED2[4:0] entry added with non-IPMI event: [Assertion|Deassertion]			
			02xxh		xxx2	Entry added with one or more SEL entries	%ED2[4:0] entry added with SEL entries: [Assertion|Deassertion]			
			03xxh		xxx3	Log cleared	%ED2[4:0] cleared: [Assertion|Deassertion]			
			04xxh		xxx4	Log disabled	%ED2[4:0] disabled: [Assertion|Deassertion]			
			05xxh		xxx5	Log enabled	%ED2[4:0] enabled: [Assertion|Deassertion]			
			other			Unknown log action	%ED2[4:0] unknown aux log action: [Assertion|Deassertion]			
						ED2 - 3:0 - Log Type	The string represented by the low nibble of ED2 is %ED2[4:0]			
			xx00h		02B0	MCA Log	MCA Auxiliary Log - %ED2[7:4]: [Assertion|Deassertion]			

Table 93. System Event Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 2 of 2)

Sensor Type	STC	OF[a]	ED2	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Event	12h	03h	xx01h		02C0	OEM 1	OEM 1 Auxiliary Log - %ED2[7:4]: [Assertion|Deassertion]	OK	OK	Yes
			xx02h		02D0	OEM 2	OEM 2 Auxiliary Log - %ED2[7:4]: [Assertion|Deassertion]			
						Reserved	Unknown Auxiliary Log - %ED2[7:4]: [Assertion|Deassertion]			
		04h	PEF Action - ED2 indicates the Action Type							
			0010 0000b		0294	Diagnostic Interrupt (NMI)	PEF Action - diagnostic interrupt (NMI): [Assertion|Deassertion]			
			0001 0000b			OEM action	PEF Action - OEM action: [Assertion|Deassertion]			
			0000 1000b			Power cycle	PEF Action - power cycle: [Assertion|Deassertion]			
			0000 0100b			Reset	PEF Action - reset: [Assertion|Deassertion]			
			0000 0010b			Power off	PEF Action - power off: [Assertion|Deassertion]			
			0000 0001b			Alert	PEF Action - alert: [Assertion|Deassertion]			
			other			Unknown PEF action	PEF Action - unknown PEF action: [Assertion|Deassertion]			

a. If more than one bit is set to 1 in the bit vector for the System Event sensor with Event Offset 04h, the strings associated with all of those bits are concatenated in the output.
b. Event Codes are in hexadecimal.
c. Throughout this table bits *m* through *n* in ED2 are denoted by %ED2[*m*:*n*].

Table 94. Critical Interrupt Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Critical Interrupt	13h	00h			02A0	Front Panel NMI / Diag Interrupt (A)	Front panel NMI/ Diagnostic interrupt: Assertion	Major	-	Yes
						Front Panel NMI / Diag Interrupt (D)	Front panel NMI/ Diagnostic interrupt: Deassertion	-	OK	Yes
		01h			02A1	Bus Timeout (A)	Bus timeout: Assertion	Major	-	Yes
						Bus Timeout (D)	Bus timeout: Deassertion	-	OK	Yes
		02h			02A2	I/O Channel check NMI (A)	I/O channel check NMI: Assertion	Major	-	Yes
						I/O Channel check NMI (D)	I/O channel check NMI: Deassertion	-	OK	Yes
		03h			02A3	SW NMI (A)	Software NMI: Assertion	Major	-	Yes
						SW NMI (D)	Software NMI: Deassertion	-	OK	Yes
		04h			02A4	PCI PERR (A)	PCI PERR detected: Assertion	Major	-	Yes
						PCI PERR (D)	PCI PERR detected: Deassertion	-	OK	Yes
		05h			02A5	PCI SERR (A)	PCI SERR detected: Assertion	Major	-	Yes
						PCI SERR (D)	PCI SERR detected: Deassertion	-	OK	Yes
		06h			02A6	EISA Fail Safe Timeout (A)	EISA fail safe timeout: Assertion	Major	-	Yes
						EISA Fail Safe Timeout (D)	EISA fail safe timeout: Deassertion	-	OK	Yes
		07h			02A7	Bug Correctable Error (A)	Bus correctable error: Assertion	Major	-	Yes
						Bug Correctable Error (D)	Bus correctable error: Deassertion	-	OK	Yes
		08h			02A8	Bus Uncorrectable Error (A)	Bus uncorrectable error: Assertion	Major	-	Yes
						Bus Uncorrectable Error (D)	Bus uncorrectable error: Deassertion	-	OK	Yes
		09h			02A9	Fatal NMI (A)	Fatal NMI: Assertion	Major	-	Yes
						Fatal NMI (D)	Fatal NMI: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 95. Button Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Button/ Switch	14h	00h			0520	Power Button pressed	Power Button pressed: [Assertion|Deassertion]	OK	OK	No
		01h			0521	Sleep Button pressed	Sleep Button pressed: [Assertion|Deassertion]	OK	OK	No
		02h			0522	Reset Button pressed	Reset Button pressed: [Assertion|Deassertion]	OK	OK	No
		03h			0523	FRU latch open	FRU latch open: [Assertion|Deassertion]	OK	OK	No
		04h			0524	FRU service request button	FRU service request button: [Assertion|Deassertion]	OK	OK	No

a. Event Codes are in hexadecimal.

Table 96. Module/Board Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Module / Board	15h	-				-	-	-	-	-

Table 97. Microcontroller/Coprocessor Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Microcontroller/ Coprocessor	16h	-				-	-			

Table 98. Add-in Card Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Add-in Card	17h	-				-	-	-	-	-

Table 99. Chassis Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Chassis	18h	-				-	-	-	-	-

Table 100. Chip Set Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Chip Set	19h	-				-	-			

Table 101. Other FRU Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Other FRU	1Ah	-				-	-			

Table 102. Cable / Interconnect Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Cable / Interconnect	1Bh	-				-	-			

Table 103. Terminator Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Terminator	1Ch	-				-	-			

Table 104. System Boot Initiated Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
System Boot Initiated	1Dh	00h			0550	Initiated by power up	Initiated by power up	OK	OK	No
		01h			0551	Initiated by hard reset	Initiated by hard reset	OK	OK	No
		02h			0552	Initiated by warm reset	Initiated by warm reset	OK	OK	No
		03h			0553	User requested PXE boot	User requested PXE boot	OK	OK	No
		04h			0554	Automated boot to diagnostic	Automated boot to diagnostic	OK	OK	No

a. Event Codes are in hexadecimal.

Table 105. Boot Error Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Boot Error	1Eh	00h			02E0	No bootable media (A)	No bootable media: Assertion	Major	-	Yes
						No bootable media (D)	No bootable media: Deassertion	-	OK	Yes
		01h			02E1	Non-bootable diskette left in drive (A)	Non-bootable diskette left in drive: Assertion	Major	-	Yes
						Non-bootable diskette left in drive (D)	Non-bootable diskette left in drive: Deassertion	-	OK	Yes
		02h			02E2	PXE Server not found (A)	PXE server not found: Assertion	Major	-	Yes
						PXE Server not found (D)	PXE server not found: Deassertion	-	OK	Yes
		03h			02E3	Invalid boot sector (A)	Invalid boot sector: Assertion	Major	-	Yes
						Invalid boot sector (D)	Invalid boot sector: Deassertion	-	OK	Yes
		04h			02E4	Timeout waiting for user selection of boot source (A)	Timeout waiting for user selection of boot source: Assertion	Major	-	Yes
						Timeout waiting for user selection of boot source (D)	Timeout waiting for user selection of boot source: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 106. OS Boot Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
OS Boot	1Fh	00h			02F0	A: boot completed	A: boot completed: [Assertion|Deassertion]	OK	OK	No
		01h			02F1	C: boot completed	C: boot completed: [Assertion|Deassertion]	OK	OK	No
		02h			02F2	PXE boot completed	PXE boot completed: [Assertion|Deassertion]	OK	OK	No
		03h			02F3	Diagnostic boot completed	Diagnostic boot completed: [Assertion|Deassertion]	OK	OK	No
		04h			02F4	CD-ROM boot completed	CD-ROM boot completed: [Assertion|Deassertion]	OK	OK	No
		05h			02F5	ROM boot completed	ROM boot completed: [Assertion|Deassertion]	OK	OK	No
		06h			02F6	boot completed - boot device not specified	boot completed - boot device not specified: [Assertion|Deassertion]	OK	OK	No

a. Event Codes are in hexadecimal.

Table 107. OS Critical Stop Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
OS Critical Stop	20h	00h			0340	Stop during OS load / init (A)	Stop during OS load/ initialization: Assertion	Major	-	Yes
						Stop during OS load / init (D)	Stop during OS load/ initialization: Deassertion	-	OK	Yes
		01h			0341	Run-time Stop (A)	Run time stop: Assertion	Major	-	Yes
						Run-time Stop (D)	Run time stop: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 108. Slot/Connector Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	(D)	SH
Slot / Connector	21h	00h			0480	Fault Status asserted	Fault Status%ED2[b]%ED3[c]: [Assertion|Deassertion]	Minor	OK	Yes
		01h			0481	Identify Status asserted	Identity Status%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		02h			0482	Slot/Connector dev installed/attached	Device Attached%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		03h			0483	Slot/Connector Ready for dev Install	Ready for Device Install%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		04h			0484	Slot/Connector ready for dev removal	Ready for Device Removal%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		05h			0485	Slot Power is Off	Connector power off%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		06h			0486	Slot/Connector dev removal request	Device removal request%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		07h			0487	Interlock asserted	Interlock%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		08h			0488	Slot is disabled	Connector disabled%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
		09h			0489	Slot holds spare device %ED2 - [6:0] Slot/ Connector Type	Connector holds spare%ED2%ED3: [Assertion|Deassertion]	OK	OK	No
			00h			- PCI	, PCI	OK	OK	No
			01h			- Drive Array	, Drive	OK	OK	No
			02h			- External Peripheral Connector	, Periph	OK	OK	No
			03h			- Docking	, Docking	OK	OK	No
			04h			- Other std internal expansion slot	, Slot	OK	OK	No
			05h			- Slot assoc w/ entity spec by Entity ID for sensor	, Entity	OK	OK	No
			06h			- ATCA	, AdvancedTCA	OK	OK	No
	21h		07h			- DIMM/memory device	, DIMM	OK	OK	No
			08h			- FAN	, FAN	OK	OK	No
				XXh		- Slot/Connector Number	0x%02x	OK	OK	No

a. Event Codes are in hexadecimal.
b. ED2 indicates slot/connector type
c. ED3 indicates slot/connector number.

Table 109. System ACPI Power State Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 1 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System ACPI Power State	22h	00h			0320	S0/G0 working (A)	ACPI State S0/G0 (working): Assertion	OK	-	Yes
						S0/G0 working (D)	ACPI State S0/G0 (working): Deassertion	-	OK	Yes
		01h			0321	S1 sleeping with hardware and processor context maintained (A)	ACPI State S1 (sleeping with hardware and processor contact maintained): Assertion	OK	-	Yes
						S1 sleeping with hardware and processor context maintained (D)	ACPI State S1 (sleeping with hardware and processor contact maintained): Deassertion	-	OK	Yes
		02h			0322	S2 sleeping, processor context lost (A)	ACPI State S2 (sleeping, processor context lost): Assertion	OK	-	Yes
						S2 sleeping, processor context lost (D)	ACPI State S2 (sleeping, processor context lost): Deassertion	-	OK	Yes
		03h			0323	S3 sleeping, processor and hardware context lost, memory retained (A)	ACPI State S3 (sleeping, h/w & processor context lost, memory retained): Assertion	OK	-	Yes
						S3 sleeping, processor and hardware context lost, memory retained (D)	ACPI State S3 (sleeping, h/w & processor context lost, memory retained): Deassertion	-	OK	Yes
		04h			0324	S4 non-volatile sleep/suspend-to-disk (A)	ACPI State S4 (non-volatile sleep, suspend to disk): Assertion	OK	-	Yes
						S4 non-volatile sleep/suspend-to-disk (D)	ACPI State S4 (non-volatile sleep, suspend to disk): Deassertion	-	OK	Yes
		05h			0325	S5 / G2 soft-off (A)	ACPI State S5/G2 (soft off): Assertion	OK	-	Yes
						S5 / G2 soft-off (D)	ACPI State S5/G2 (soft off): Deassertion	-	OK	Yes
		06h			0326	S4 / S5 soft-off, particular S4/S5 state can't be deter (A)	ACPI State S4/S5 soft-off: Assertion	OK	-	Yes
						S4 / S5 soft-off, particular S4/S5 state can't be deter (D)	ACPI State S4/S5 soft-off: Deassertion	-	OK	Yes
		07h			0327	G3 / Mechanical Off (A)	ACPI State G3/Mechanical Off: Assertion	OK	-	Yes
						G3 / Mechanical Off (D)	ACPI State G3/Mechanical Off: Deassertion	-	OK	Yes

Table 109. System ACPI Power State Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 2 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System ACPI Power State	22h	08h			0328	Sleeping in S1, S2, or S3 states (A)	ACPI State (Sleeping in an S1, S2 or S3 state): Assertion	OK	-	Yes
						Sleeping in S1, S2, or S3 states (D)	ACPI State (Sleeping in an S1, S2 or S3 state): Deassertion	-	OK	Yes
		09h			0329	G1 sleeping (A)	ACPI State G1 sleeping: Assertion	OK	-	Yes
						G1 sleeping (D)	ACPI State G1 sleeping: Deassertion	-	OK	Yes
		0Ah			032A	S5 entered by override (A)	ACPI State S5 entered by override: Assertion	OK	-	Yes
						S5 entered by override (D)	ACPI State S5 entered by override: Deassertion	-	OK	Yes
		0Bh			032B	Legacy ON state (A)	ACPI legacy ON state: Assertion	OK	-	Yes
						Legacy ON state (D)	ACPI legacy ON state: Deassertion	-	OK	Yes
		0Ch			032C	Legacy OFF state (A)	ACPI legacy OFF state: Assertion	OK	-	Yes
						Legacy OFF state (D)	ACPI legacy OFF state: Deassertion	-	OK	Yes
		0Eh			032D	Unknown (A)	ACPI state unknown: Assertion	OK	-	Yes
						Unknown (D)	ACPI state unknown: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 110. Watchdog 2 Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 1 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Watchdog 2	23h	00h			0350	Timer expired (A)	Timer expired status only%ED2[b]: Assertion	OK	-	No
						Timer expired (D)	Timer expired status only%ED2: Deassertion	-	OK	No
		01h			0351	Hard Reset (A)	Hard reset%ED2: Assertion	OK	-	No
						Hard Reset (D)	Hard reset%ED2: Deassertion	-	OK	No
		02h			0352	Power Down (A)	Power down%ED2: Assertion	OK	-	No
						Power Down (D)	Power down%ED2: Deassertion	-	OK	No
		03h			0353	Power Cycle (A)	Power cycle%ED2: Assertion	OK	-	No
						Power Cycle (D)	Power cycle%ED2: Deassertion	-	OK	No
		04h - 07h				reserved		-	-	-
		08h			0354	Timer interrupt (A)	Timer interrupt generated%ED2: Assertion	OK	-	No
						Timer interrupt (D)	Timer interrupt generated%ED2: Deassertion	-	OK	No
		%ED2 in the “Timer interrupt generated” string is replaced by one of the interrupt types below.								
			00xxh			None	, Non-interrupt timer	-	-	No
			01xxh			SMI	, SMI interrupt type	-	-	No
			02xxh			NMI	, NMI interrupt type	-	-	No
			03xxh			Messaging Interrupt	, Messaging interrupt type	-	-	No
			0Fxxh			unspecified	, Unspecified interrupt type	-	-	No
			xx00h			reserved		-	-	No
			xx01h			BIOS/FRB2	, BIOS FRB2 timer	-	-	No
			xx02h			BIOS/POST	, BIOS/POST timer	-	-	No
			xx03h			OS Load	, OS Load timer	-	-	No

Table 110. Watchdog 2 Sensor from IPMI 1.5 Spec, Table 36-3 (sheet 2 of 2)

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Watchdog 2	23h		xx04h			SMS/OS	, SMS/OS timer	-	-	No
			xx05h			OEM	, OEM timer	-	-	No
			xx0Fh			unspecified	, Unspecified timer	-	-	No

a. Event codes are in hexadecimal.
b. ED2 provides an event extension code using the definitions from the IPMI v1.5 Specification.

Table 111. Platform Alert Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Platform Alert	24h	00h			0380	Platform generated page	Platform generated page: [Assertion|Deassertion]	OK	OK	No
		01h			0381	Platform generated LAN alert	Platform generated LAN alert: [Assertion|Deassertion]	OK	OK	No
		02h			0382	Platform event trap generated	Platform Event Trap generated: [Assertion|Deassertion]	OK	OK	No
		03h			0383	Platform generated SNMP trap, OEM format	Platform generated SNMP trap, OEM format: [Assertion|Deassertion]	OK	OK	No

a. Event Codes are in hexadecimal.

Table 112. Entity Presence Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Entity Presence	25h	00h			0390	Entity Present	Entity Present: [Assertion|Deassertion]	OK	Major	Yes[b]
		01h			0391	Entity Absent	Entity Absent: [Assertion|Deassertion]	Major	OK	Yes
		02h			0392	Entity Disabled	Entity Disabled: [Assertion|Deassertion]	Major	OK	Yes

a. Event Codes are in hexadecimal.
b. Presence Sensors on PEMs, Fans, Filter Trays, Shelf FRU contribute to system health.

Table 113. Monitor ASIC/IC Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Monitor ASIC / IC	26h	-				-	-			

Table 114. LAN Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
LAN	27h	00h			0050	LAN Heartbeat Lost (A)	LAN Heartbeat Lost: Assertion	Minor	-	Yes
					0051	LAN Heartbeat Lost (D)	LAN Heartbeat Lost: Deassertion	-	OK	Yes
		01h			0052	LAN Heartbeat (A)	LAN Heartbeat: Assertion	OK	-	Yes
					0053	LAN Heartbeat (D)	LAN Heartbeat: Deassertion	-	Minor	Yes
		02h			0054	Duplicate IP Address detected (A)	Duplicate IP address detected: Assertion	Major	-	Yes
					0055	Duplicate IP Address detected (D)	Duplicate IP address detected: Deassertion	-	OK	Yes

a. Event Codes are in hexadecimal.

Table 115. Management Subsystem Health Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Management Subsystem Health	28h	00h			0500	sensor access degraded or unavailable	sensor access degraded or unavailable: [Assertion|Deassertion]	Minor	OK	Yes
		01h			0501	controller access degraded or unavailable	controller access degraded or unavailable: [Assertion|Deassertion]	Minor	OK	Yes
		02h			0502	management controller off-line	management controller off-line: [Assertion|Deassertion]	Major	OK	Yes
		03h			0503	management controller unavailable	management controller unavailable: [Assertion|Deassertion]	Major	OK	Yes

a. Event Codes are in hexadecimal.

Table 116. Battery Sensor from IPMI 1.5 Spec, Table 36-3

Sensor Type	STC	OF	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Battery	29h	00h			0530	battery low (predictive failure)	battery low (predictive failure): [Assertion|Deassertion]	Minor	OK	Yes
		01h			0531	battery failed	battery failed: [Assertion|Deassertion]	Major	OK	Yes
		02h			0532	battery presence detected	battery presence detected: [Assertion|Deassertion]	OK	OK	Yes

a. Event Codes are in hexadecimal.

Appendix D OEM Sensor Events

D.1 Introduction

This appendix lists all of the OEM sensors and events defined by Radisys for the A6K-RSM-J shelf manager module. These events are defined in accordance with the IPMI Specification version 1.5.

D.2 Explanation of Abbreviations and Symbols

This section explains the column heading abbreviations and special symbols used in the tables in this appendix.

- STC means Sensor Type Code
- ERC means Event Reading Code
- OF means Sensor-specific Offset
- ED2 means Event Data 2
- ED3 means Event Data 3
- EC means Event code (in hexadecimal notation)
- SH means System Health contribution
- (A) means Assertion
- (D) means Deassertion
- Dash (–) means “not applicable”.

D.3 PICMG Hot Swap Sensor

Table 117. PICMG Hot Swap Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Hot Swap	F0h	6Fh	00h			130h	Hot Swap State Change	FRU %1 transitioned from %2 to %3 %4 where, %1 = FRU ID from ED3 %2 = Old State from ED2[3:0], %3 = New State from Offset %4 = Change Cause from ED2[7:4] For possible values of %2 & %3 see Table 118, “Hot Swap States” on page 246 For possible values of %4 see Table 119, “Hot Swap State Change Cause” on page 246			no
			01h			131h					no
			02h			132h					no
			03h			133h					no
			04h			134h					no
			05h			135h					no
			06h			136h					no
			07h			137h			Major	OK	yes
			08h			13Eh			Major	OK	yes
			09h						Major	OK	yes
			0Ah						Major	OK	yes
			0Bh						Major	OK	yes
			0Ch						Major	OK	yes
			0Dh						Major	OK	yes
			0Eh						Major	OK	yes
			0Fh						Major	OK	yes
			00h	8xh	ED3	13Fh		Invalid hardware address %1 detected where, %1 = HW address from ED3	Major	OK	yes

Note: In specific situations, the RSM may generate a Hot Swap event with the sensor number set to 0xFF (RESERVED). Such events are generated to signal M-state transitions for FRUs for which SDR records are not available yet. Currently, Hot Swap events with sensor number set to 0xFF are generated by the RSM in the following situations:

- RSM receives a non-Hot Swap event from a FRU whose M-state is not known to the RSM
- RSM detects an unknown FRU during the E-keying process

Table 118. Hot Swap States

Code	Description
00h	Not Installed (M0)
01h	Inactive (M1
02h	Activation Request (M2)
03h	Activation In Progress (M3)
04h	Active (M4)
05h	Deactivation Request (M5)
06h	Deactivation In Progress (M6)
07h	Communication Lost (M7)
08h-0Fh	Reserved (%02Xh)

Table 119. Hot Swap State Change Cause

Code	Description
00h	Due to Normal State Change
01h	Due to Command by Shelf Manager with Set FRU Activation
02h	Due to Operator changing the handle switch
03h	Due to Programmatic action
04h	Due to Communication Failure
05h	Due to Communication Failure caused by Local Malfunction
06h	Due to Surprise Extraction
07h	Due to Information Provided by user/System
08h	Due to Invalid Hardware Address
09h	Due to Unexpected Deactivation
0Fh	Cause Unknown

D.4 PICMG IPMB-0 Link Sensor

Table 120. PICMG IPMB-0 Link Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
IPMB-0 Link State	F1h	6Fh	00h			140h	IPMB-0 Link State Change	IPMB-%1 changed state to %2 - IPMB-A state is %3, %4 - IPMB-B state is %5, %6 where %1 = IPMB Channel Number from ED2[7:4] %2 = IPMB Link State from Offset %3 = IPMB Link Local Control State for IPMB-A from ED3[3] %4 = IPMB Link State Event for IPMB-A from ED3[2:0] %5 = IPMB Link Local Control State for IPMB-A from ED3[7] %6 = IPMB Link State Event for IPMB-A from ED3[6:4] For possible values of %2 see Table 121, "IPMB Link State" on page 247 For possible values of %3 and %5 see Table 122, "IPMB Link Local Control State" on page 247 For possible values of %4 and %6 see Table 123, "IPMB Link State Event" on page 248	Major	OK	yes
			01h			141h			Major	OK	yes
			02h			142h			Major	OK	yes
			03h			143h			OK	Major	yes

Table 121. IPMB Link State

Code	Description
00h	IPMB-A disabled, IPMB-B disabled
01h	IPMB-A enabled, IPMB-B disabled
02h	IPMB-A disabled, IPMB-B enabled
03h	IPMB-A enabled, IPMB-B enabled

Table 122. IPMB Link Local Control State

Code	Description
00h	Isolated
01h	Local Control State

Table 123. IPMB Link State Event

Code	Description
00h	No failure
01h	Unable to drive clock line high
02h	Unable to drive data line high
03h	Unable to drive clock line low
04h	Unable to drive data line low
05h	Clock low timeout
06h	Under test
07h	Undiagnosed communications failure

D.5 HA Trap Connect Sensor

Table 124. HA Trap Connect Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA Trap Connect	C5h	70h	00h			1100	Trap Address 1 connectivity	Trap address 1 not responding or not configured	Major[a]	OK	yes

a. This event has assertion severity at Major level but its health score contribution is at Critical level.

D.6 HA Out of Service Request Sensor

Table 125. HA Out of Service Request Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA Out of Service Request	DCh	70h	00h			1120	Out-of-service user command	Out-of-service user command			no
			02h			1122	IPMB-0 lost	IPMB-0 lost			no
			03h			1123	M1 transition request (Deactivate FRU)	M1 transition request (Deactivate FRU)			no
			04h			1124	Shutdown request (SIGTERM)	Shutdown request (SIGTERM)			no
			05h			1125	Active HW state seized	Active HW state seized			no
			06h			1126	No active nor standby role assigned in the election	No active nor standby role assigned in the election			no
			07h			1127	Shelf FRU election failed	Shelf FRU election failed			no
			08h			1128	IP connectivity lost on a standby CMM	IP connectivity lost on a standby CMM			no
			09h			1129	Chassis detection failed	Chassis detection failed			no
			0Ah			112A	Process Monitoring graceful reboot request	Process Monitoring graceful reboot request			no
			0Bh			112B	Process Monitoring reboot request	Process Monitoring reboot request			no
			0Ch			112C	FRU control IPMI request (Deactivate)	FRU control IPMI request (Deactivate)			no
			0Dh			112D	IPMC not ready	IPMC not ready			no
			0Eh			112E	Invalid license	Invalid license			no

D.7 HA In Service Request Sensor

Table 126. HA In Service Request Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA In Service Request	DDh	70h	00h			1140	In-service user command	In-service user command			no
			01h			1141	Ejector closed request	Ejector closed request			no
			02h			1142	IPMB-0 recovered	IPMB-0 recovered			no
			03h			1143	FRU activate IPMI request	FRU activate IPMI request			no
			04h			1144	IPMC Ready	IPMC Ready			no

D.8 HA State Sensor

Table 127. HA State Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA State	C9h	70h	00h			1150	Out-of-service readiness state	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253 Note: this is the default output			no
								Current state: %1; Previous readiness and HA state: %2; Reason to enter the out-of-service state %3 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] %3 = Reason to enter OOS state from ED2[3:0] For possible values of %1 & %2 see Table 128, "Readiness and HA State Codes" on page 253 For possible values of %3 see Table 129, "Reasons to Enter OOS State" on page 253 Note: this output applies only to the transition from the election state to the out-of-service state, i.e. Offset=0, ED2[7:4]=1			
			01h			1151	Election readiness state	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253			no

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA State	C9h	70h	02h			1152	In-service readiness state; active-no-standby	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253 Note: this is the default output			no
								Current state: %1; Previous state: %2; Peer disconnection indication %3 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] %3 = Peer disconnection indication from ED2[3:0] For possible values of %1 & %2 see Table 128, "Readiness and HA State Codes" on page 253 For possible values of %3 see Table 130, "Peer Disconnection Indication" on page 253 Note: this output applies only to the transition from the active or standby state to the active-no-standby state, i.e. Offset=2, ED2[7:4]=5 or ED2[7:4]=3			
			03h			1153	In-service readiness state; active	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253			no
HA State	C9h	70h	04h			1154	In-service readiness state; quiesced	Current state: %1; Previous state: %2; Reasons to enter quiesced state %3 %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] %3 = Reason to enter quiesced state from ED2[3:0] For possible values of %1 & %2 see Table 128, "Readiness and HA State Codes" on page 253 For possible values of %3 see Table 131, "Reason to enter quiesced state" on page 253			no

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA State	C9h	70h	05h			1155	In-service readiness state; standby	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253			no
			06h			1156	In-service readiness; stopping	Current state: %1; Previous state: %2 where %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] For possible values of %1 and %2 see Table 128, "Readiness and HA State Codes" on page 253 Note: this is the default output			no
								Current state: %1; Previous state: %2; Reason to enter stopping state %3 %1 = Current HA state from Offset %2 = Previous HA state from ED2[7:4] %3 = Reason to enter stopping state For possible values of %1 & %2 see Table 128, "Readiness and HA State Codes" on page 253 For possible values of %3 see Table 129, "Reasons to Enter OOS State" on page 253 Note: this output applies only to the transition from the active, standby or active-no-standby state to the stopping state, i.e. Offset=6, ED2[7:4]=4 or ED2[7:4]=5 or ED2[7:4]=2			

Table 128. Readiness and HA State Codes

Code	Description
00h	out-of-service readiness state
01h	election readiness state
02h	in-service readiness state: active-no-standby HA state
03h	in-service readiness state: active HA state
04h	in-service readiness state: quiesced HA state
05h	in-service readiness state: standby HA state
06h	in-service readiness state: stopping HA state

Table 129. Reasons to Enter OOS State

Code	Description
00h	out-of-service request
01h	IP connection lost (for elected standby only)
02h	no-role assigned in election/ active and standby already present
03h	shelf FRU election failed

Table 130. Peer Disconnection Indication

Code	Description
00h	indication not available
01h	HW presence or health signal
02h	peer in-service exit message received
03h	IPMB-0 keep alive not received
04h	IP connectivity lost

Table 131. Reason to enter quiesced state

Code	Description
00h	switchover (health change)
01h	manual switchover
02h	out-of-service request

Table 132. Reason to enter stopping state

Code	Description
00h	out-of-service request
01h	IP connection lost (for standby state only)

D.9 DataSync Status Sensor

Table 133. DataSync Status Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
DataSync Status	DEh	70h	00h			1160	Data Synchronization running	Data Synchronization running			no
			01h			1161	Priority 1 Data is synced	Priority 1 Data is synced			no
			02h			1162	Priority 2 Data is synced	Priority 2 Data is synced			no
			03h			1163	Initial Data Synchronization complete	Initial Data Synchronization complete			no

D.10 HA Health Score Sensor

Table 134. HA Health Score Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Health Score	D3h	70h	00h			1170	Critical health score change occurred on this CMM	Critical health score change occurred on this CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no
			01h			1171	Major health score change occurred on this CMM	Major health score change occurred on this CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no
			02h			1172	Minor health score change occurred on this CMM	Minor health score change occurred on this CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no
Health Score	D3h		03h			1173	Critical health score change occurred on other CMM	Critical health score change occurred on other CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no
			04h			1174	Major health score change occurred on other CMM	Major health score change occurred on other CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no
			05h			1175	Minor health score change occurred on other CMM	Minor health score change occurred on other CMM: New health score value %1 previous health score value %2 where %1 = health score from ED2[7:0] %2 = health score from ED3[7:0]			no

D.11 HA Redundancy Sensor

Table 135. HA Redundancy Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA Redundancy	C8h	70h	00h			1180	Not operational	Not operational			no
			01h			1181	Proposed active role; shelf FRU election	Proposed active role; shelf FRU election; Peer disconnection indication %1 where %1 = Peer disconnection indication from ED2[3:0] For possible values of %1 see Table 130, "Peer Disconnection Indication" on page 253			no
			02h			1182	Sending IP configuration to elected standby	Sending IP configuration to elected standby			no
			03h			1183	Connecting over IP	Connecting over IP			no
			04h			1184	Sending shelf FRU and configuration to elected standby	Sending shelf FRU and configuration to elected standby			no
			05h			1185	Operational/In-service	Operational/In-service; Peer disconnection indication %1 where %1 = Peer disconnection indication from ED2[3:0] For possible values of %1 see Table 130, "Peer Disconnection Indication" on page 253			no
			06h			1186	Proposed standby role	Proposed standby role; waiting for shelf FRU result			no
HA Redundancy			07h			1187	Receiving IP configuration from active	Receiving IP configuration from active			no
			08h			1188	Receiving shelf FRU and configuration from active	Receiving shelf FRU and configuration from active			no
			09h			1189	Disconnecting	Disconnecting.			no
			0Ah			118A	Local shelf FRU election failed	Local shelf FRU election failed. Waiting for shelf FRU result on peer			no
			0Bh			118B	Unknown shelf detected	Unknown shelf detected. Waiting for shelf FRU election result on peer			no
			0Ch			118C	IP configuration initialization	IP configuration initialization			no

D.12 HA Control Sensor

Table 136. HA Control Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA Control	D2h	70h	00h			1200	HA Control event	HA control event: %1 where %1 = HA Control event type from ED2[6:0] For possible values of %1 see Table 137, "HA Control Event Type" on page 257			no
			01h			1201	Peer in-service exit message	Peer in-service exit message %1 received where %1 = Peer in service exit reason from ED2[3:0] For possible values of %1 see Table 138, "Peer in service exit reason" on page 258			no

Table 137. HA Control Event Type

Code	Description
00h	out-of-service request
01h	peer out-of-service request
02h	remote out-of-service request
03h	in-service request
04h	peer in-service request
05h	remote in-service request received
06h	peer forced exit request
07h	manual switchover request
08h	peer manual switchover request
09h	remote manual switchover request
0Ah	automatic switchover request
0Bh	deactivate FRU IPMI message request
0Ch	activate FRU IPMI message request
0Dh	process monitoring reboot request
0Eh	process monitoring graceful reboot request
0Fh	FRU control IPMI message request (deactivate)
10h	Standby reboot request
11h	Remote standby reboot request received

Table 138. Peer in service exit reason

Code	Description
00h	out-of-service user command
02h	IPMB-0 lost
03h	M1 transition request (Deactivate FRU)
04h	shutdown request (SIGTERM)
05h	active HW state seized
06h	no active nor standby role assigned in the election
07h	shelf FRU election failed
08h	IP connectivity lost on a standby CMM
09h	chassis detection failed
0Ah	process monitoring graceful reboot request
0Bh	process monitoring reboot request
0Ch	FRU control IPMI request (Deactivate)

D.13 PMS Fault Sensor

Table 139. PMS Fault Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
PMS Fault	DAh	07h	00h			170h	Process existence fault; attempting recovery	PmsProc%1\t[a]Process existence fault; attempting recovery where %1 = Process unique ID from ED3	see note	see note	yes
			01h			171h	Process integrity fault; attempting recovery	PmsProc%1\tProcess integrity fault; attempting recovery where %1 = Process unique ID from ED3	see note	see note	yes
			02h			172h	Thread watchdog fault; attempting recovery	PmsProc%1\tThread watchdog fault; attempting recovery where %1 = Process unique ID from ED3	see note	see note	yes
			03h			173h	Process existence fault; monitoring disabled	PmsProc%1\tProcess existence fault; monitoring disabled where %1 = Process unique ID from ED3	see note	see note	yes
			04h			174h	Process integrity fault; monitoring disabled	PmsProc%1\tProcess integrity fault; monitoring disabled where %1 = Process unique ID from ED3	see note	see note	yes
			05h			175h	Thread watchdog fault; monitoring disabled	PmsProc%1\tThread watchdog fault; monitoring disabled where %1 = Process unique ID from ED3	see note	see note	yes
			06h			176h	Excessive reboots/ failovers; all process monitoring disabled	PmsProc%1\tExcessive reboots/ failovers; all process monitoring disabled where %1 = Process unique ID from ED3	see note	see note	yes
			07h			177h	Recovery successful	PmsProc%1\tRecovery successful where %1 = Process unique ID from ED3	see note	see note	yes
			08h			178h	Monitoring initialized	PmsProc%1\tMonitoring initialized where %1 = Process unique ID from ED3	see note	see note	yes

a. \t indicates a Tab character

Note: Event severity is set in the high nibble of ED2 following the event severity states from generic reading type 07h. (See Table 36-2 in the IPMI 1.5 Specification.) 0 = OK, 1 = minor, 2 = major, 3 = critical, 4 = minor, 5 = major, 6 = critical, 7 = OK, 8 = OK

D.14 PMS Info Sensor

Table 140. PMS Info Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A) (D)		SH
PMS Info	DBh	70h	00h			179h	Take no action specified for recovery	PmsProc%1\t[a]Take no action specified for recovery where %1 = Process unique ID from ED3			no
			01h			17Ah	Attempting process restart recovery action	PmsProc%1\tAttempting process restart recovery action where %1 = Process unique ID from ED3			no
			02h			17Bh	Attempting process failover & restart recovery action	PmsProc%1\tAttempting process failover & restart recovery action where %1 = Process unique ID from ED3			no
			03h			17Ch	Attempting process failover & reboot recovery action	PmsProc%1\tAttempting process failover & reboot recovery action where %1 = Process unique ID from ED3			no
			04h			17Dh	Take no action specified for escalated recovery	PmsProc%1\tTake no action specified for escalated recovery where %1 = Process unique ID from ED3			no
			05h			17Eh	Attempting process failover & restart escalated recovery action	PmsProc%1\tAttempting process failover & restart escalated recovery action where %1 = Process unique ID from ED3			no
			06h			17Fh	Process restart recovery failure	PmsProc%1\tProcess restart recovery failure where %1 = Process unique ID from ED3			no
			07h			180h	Failover & reboot recovery failure	PmsProc%1\tFailover & reboot recovery failure where %1 = Process unique ID from ED3			no
			08h			181h	Recovery failure due to excessive restarts	PmsProc%1\tRecovery failure due to excessive restarts where %1 = Process unique ID from ED3			no
			09h			182h	Failover & reboot escalated recovery failure	PmsProc%1\tFailover & reboot escalated recovery failure where %1 = Process unique ID from ED3			no
			0Ah			183h	Internal fault detected; monitoring disabled	PmsProc%1\tInternal fault detected; monitoring disabled where %1 = Process unique ID from ED3			no

a. \t indicates a Tab character

D.15 PMS Health Sensor

Table 141. PMS Health Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
PMS Health	C7h	70h	00h			12C0	Minor events exists	Minor events exists for PmsProc%1 where %1 = Process unique ID from ED3	Minor	OK	yes
			01h			12C1	Major events exists	Minor events exists for PmsProc%1 where %1 = Process unique ID from ED3	Major	OK	yes
			02h			12C2	Critical events exists	Minor events exists for PmsProc%1 where %1 = Process unique ID from ED3	Critical	OK	yes

D.16 Local Upgrade Sensor

Table 142. Local Upgrade Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Local Upgrade	DFh	70h	00h			1220	New Image Loaded	New Image Loaded; Partition %1 changed; OS Loader has %2been upgraded; Linux kernel has %3been upgraded; Root fs has %4been upgraded; Old Image Boot Role: %5; New Image Boot Role: %6 where %1 = Upgraded Partition Indicator from ED2[7] %2 = Not set from ED2[6] %3 = Not set from ED2[5] %4 = Not set from ED2[4] %5 = Old Image Boot Role from ED3[3:0] %6 = New Image Boot Role ED3[7:4] For possible values of %1 see Table 143, “Upgraded Partition Indicator” on page 263 For possible values of %2, %3, %4 see Table 144, “Not Set Values” on page 264 For possible values of %5, %6 see Table 145, “Image Boot Role” on page 264			no
			01h			1221	New Image Startup Success	New Image Startup Success;			no

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
			02h			1222	New Image Startup Failure	New Image Startup Failure; Partition %1 changed; Old Image Boot Role: %2; New Image Boot Role: %3 where %1 = Upgraded Partition Indicator from ED2[7] %2 = Old Image Boot Role from ED3[3:0] %3 = New Image Boot Role ED3[7:4] For possible values of %1 see Table 143, "Upgraded Partition Indicator" on page 263 For possible values of %2, %3 see Table 145, "Image Boot Role" on page 264			no
			03h			1223	Image Boot Role Changed	Image Boot Role Changed; Partition %1 changed; Old Image Boot Role: %2; New Image Boot Role: %3 where %1 = Upgraded Partition Indicator from ED2[7] %2 = Old Image Boot Role from ED3[3:0] %3 = New Image Boot Role ED3[7:4] For possible values of %1 see Table 143, "Upgraded Partition Indicator" on page 263 For possible values of %2, %3 see Table 145, "Image Boot Role" on page 264			no
			04h			1224	Active Image Partition Duplication	Active Image Partition Duplication; Partition %1 changed; Old Image Boot Role: %2; New Image Boot Role: %3 where %1 = Upgraded Partition Indicator from ED2[7] %2 = Old Image Boot Role from ED3[3:0] %3 = New Image Boot Role ED3[7:4] For possible values of %1 see Table 143, "Upgraded Partition Indicator" on page 263 For possible values of %2, %3 see Table 145, "Image Boot Role" on page 264			no

Table 143. Upgraded Partition Indicator

Code	Description
00h	A
01h	B

Table 144. Not Set Values

Code	Description
00h	not
01h	

Table 145. Image Boot Role

Code	Description
00h	default
01h	fallback
02h	one shot
03h	empty

D.17 Log Usage Sensor

Table 146. Log Usage Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Event Logging Disabled	10h						See Table 92, "Event Logging Disabled Sensor from IPMI 1.5 Spec, Table 36-3" on page 230				yes

D.18 Power Allocation Sensor

Table 147. Power Allocation Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Power Allocation	CCh	6Fh	00h			1240	Power allocation failed	Power allocation failed for FRU %1 Device ID %2 where %1 = FRU hardware address from ED2 %2 = FRU Device ID from ED3			no
			01h			1241	Power allocation completed	Power allocation completed for FRU %1 Device ID %2 where %1 = FRU hardware address from ED2 %2 = FRU Device ID from ED3			no

D.19 Power Budget Sensor

Power Budget sensors are threshold type sensors that track power budget on the RSM. There is one power budget sensor per each power feed (maximum number is 16). The sensor supports Upper Non-Recoverable, Upper Critical, and Upper Non-Critical thresholds set to 100%, 95%, and 75% of power allowance, respectively.

Table 148. Power Budget Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Power Budget	CDh	01h					See Table 77, "Generic Sensors from IPMI v1.5 Table 36-2" on page 216				no

D.20 Cooling Policy Sensor

Table 149. Cooling Policy Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Cooling Policy	CAh	6Fh	00h			12D0	Cooling policy in normal state	Cooling policy in normal state			no
			01h			12D1	Cooling policy in abnormal state	Cooling policy in abnormal state			no
			02h			12D2	Cooling policy in delay state	Cooling policy in delay state			no

D.21 Temperature Condition Sensor

Table 150. Temperature Condition Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Temperature Condition	CEh	6Fh	00h			1250	Normal temperature condition	Normal temperature condition			no
			01h			1251	Minor temperature condition	Minor temperature condition			no
			02h			1252	Major temperature condition	Major temperature condition			no
			03h			1253	Critical temperature condition	Critical temperature condition			no

D.22 Re-enumeration Sensor

Table 151. Re-enumeration Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Re-enumeration	CFh	6Fh	00h			1260	Re-enumeration completed	Re-enumeration completed; Number of detected FRUs %1 where %1 = number of detected FRUs from ED3			no
			01h			1261	Re-enumeration started	Re-enumeration started			no

D.23 RT Diagnostics Sensor

Table 152. RT Diagnostics Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
RT Diagnostics	C2h	6Fh	00h			1270	Diagnostics test flash failure	Diagnostics test flash failure; Error code %1 where %1 = Runtime Diagnostics Error code from ED3 For possible values of ED3 see Table 153, "Runtime Diagnostics Error Code" on page 268			no
			01h			1271	Diagnostics test Eth failure	Diagnostics test Eth failure; Error code %1 where %1 = Runtime Diagnostics Error code from ED3 For possible values of ED3 see Table 153, "Runtime Diagnostics Error Code" on page 268			no
			02h			1272	Diagnostics test IPMB failure	Diagnostics test IPMB failure; Error code %1 where %1 = Runtime Diagnostics Error code from ED3 For possible values of ED3 see Table 153, "Runtime Diagnostics Error Code" on page 268			no
			03h			1273	Diagnostics test LED failure	Diagnostics test LED failure; Error code %1 where %1 = Runtime Diagnostics Error code from ED3 For possible values of ED3 see Table 153, "Runtime Diagnostics Error Code" on page 268			no
			07h			1274	Diagnostics test flash executed	Diagnostics test flash executed			no
			08h			1275	Diagnostics test Eth executed	Diagnostics test Eth executed			no
			09h			1276	Diagnostics test IPMB executed	Diagnostics test IPMB executed			no
			0Ah			1277	Diagnostics test LED executed	Diagnostics test LED executed			no

Table 153. Runtime Diagnostics Error Code

Code	Description
00h	Invalid Address Error
01h	Invalid Data Error
02h	No Response Error
03h	IPMB Driver Error
04h	PMB Invalid Link Error
05h	IPMB Setting Clock Line High Error
06h	IPMB Setting Clock Line Low Error
07h	IPMB Setting Data Line High Error
08h	IPMB Setting Data Line Low Error
09h	IPMB Clock Low Error
0Ah	Unknown Error

D.24 Reboot Reason Sensor

Table 154. Reboot Reason Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Reboot Reason	C4h	70h	00h	00h		1280	Reboot	Reboot upgrade			no
				01h				Reboot manual reset			no
				02h				Reboot FRU control reset			no
				03h				Reboot PM reset			no
				04h				Reboot OS shutdown			no
				05h				Reboot kernel panic			no
				10h				Reboot undetermined none present			no
				11h				Reboot undetermined multiple present			no

D.25 Security Sensor

Table 155. Security Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Security	E0h	70h	01h			1291	Authentication failure event	Authentication failure event; Channel type %1 where %1 = Channel Type from ED3 For possible values of %1 see			no
			02h			1292	Root user password reset	Root user password reset			no

Table 156. Channel Type Codes

Code	Description
00h	SNMP
01h	RMCP
02h	Console

D.26 NTP Status Sensor

Table 157. NTP Status Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
NTP Status	C6h	70h	01h			12A1	A time server is lost	A time server is lost (not primary time server); Server index %1 where %1 = Server Index from ED3			no
			02h			12A2	The primary time server is lost	The primary time server is lost; Number of outstanding servers %1 where %1 = number of outstanding servers from ED3			no
			03h			12A3	Time synchronization is lost	Time synchronization is lost			no

D.27 Non Compliant FRU Sensor

Table 158. Non Compliant FRU Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Non Compliant FRU	CBh	70h	00h			12B0	Unspecified reason	Unspecified reason; FRU HW address %1; FRU Device ID %2 where %1 = FRU hardware address from ED2 %2 = FRU Device ID from ED3			no
			01h			12B1	Invalid transition detected	Invalid transition detected; FRU HW address %1; FRU Device ID %2 where %1 = FRU hardware address from ED2 %2 = FRU Device ID from ED3			no
			02h			12B2	Invalid state detected	Invalid state detected; FRU HW address %1; FRU Device ID %2 where %1 = FRU hardware address from ED2 %2 = FRU Device ID from ED3			no

D.28 Filter Run Time Sensor

The Filter Run Time sensor is a chassis sensor that tracks the number of days that the air filter has been installed. It supports the Upper Critical threshold that should be set to the maximum number of days that the air filter can remain installed before it must be replaced. It also supports the Upper Non-Critical threshold which can be set to n days less than the Upper Critical threshold to give advance warning that the air filter needs to be replaced in n days.

The availability of the Filter Run Time sensor depends on the chassis type.

Table 159. Filter Run Time Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Filter Run Time	C0h	01h					See Table 77, "Generic Sensors from IPMI v1.5 Table 36-2" on page 216				no

D.29 CMM Status Sensor

The CMM Status Sensor is a discrete sensor that indicates whether or not the RSM is fully up and running. The sensor uses bits of the bit vector to indicate status as shown in Table 160, "CMM Status Sensor Bits".

Table 160. CMM Status Sensor Bits

Bit Number	Bit Name	Description
0	Running	Set when the Active/Standby election of the RSMs has taken place. Reset when the RSM enters stopping or out-of-service state.
1	Active	Set when the RSM is active.
2	Enumeration	Set when the re-enumeration has finished
3	Wrapper	Set when the RSM becomes active or standby
4	SNMP	Sen when the SNMP daemon's tables are initially populated
14	Timeout	Set when the RSM exceeds a timeout waiting to become ready

The Running bit is used to be sure the Active/Standby election has taken place and the remaining status bits are valid. All bits are initialized to 0 on RSM startup and Running is set to 1 by the election process. The Running bit is cleared when RSM goes to stopping or out-of-service Readiness state.

When the active election has taken place, the RSM transitions to either active or standby state. This transition either sets (if the resulting HA state is active) or clears (if the resulting HA state is standby) the Active bit and logs either the CMM Status Active or CMM Status Standby (respectively) in the SEL. The SEL events trigger SNMP traps and launch any associated EventAction scripts.

The Enumeration bit is set by re-enumeration.

The Wrapper bit is supported for backward compatibility. It is set automatically when the RSM becomes active or standby.

The SNMP bit is set when the SNMP daemon's tables are initially populated.

If a timeout value has been set and this process takes longer than the timeout, the TIMEOUT bit is set. It is cleared once all the other status bits are set and the RSM is ready. The `cmmreadytimeout` dataitem is used to set the timeout (see "Alert Standard Format (ASF) Specification version 2.0"). The timer value is read and set when the election state is entered.

When the RSM goes to standby all bits except for Running are cleared.

When queried for its current value, the sensor displays the status bits and a textual interpretation.

For example, for an active RSM:

```
bash# cmmget -t "0:CMM Status" -d current
The current value is 0x001f
CMM Status Active
CMM enumeration is completed
CMM Status Ready
```

For the standby CMM, the output would look like this:

```
bash# cmmget -t "0:CMM Status" -d current
The current value is 0x0001
CMM is Standby
```

The final example is:

```
bash# cmmget -t "0:CMM Status" -d current
The current value is 0x0000

CMM Status is not Active nor Standby
```

These outputs reflect the status bits in the CMM Status Sensor. When the RSM has status Not Ready, information about which blades are not yet running is also displayed. As with other RSM sensor data, this item can be queried on the standby RSM.

This sensor sends events when the RSM changes status from active to standby or from standby to active, when the RSM is fully ready, or if the RSM has taken too long to become ready (by taking more time than specified in the `CMMStatusReadyTimeout` configuration parameter).

Table 161. CMM Status Sensor Format

Byte	Data Field
1	*Event Message Rev* = 04h (IPMI 1.5)
2	*Sensor Type* = D9h
3	*Sensor Number* = E8h
4	Event Direction (bit 7) = 0b (assertion) OR 1b (deassertion) Event Type [6:0] = 6Fh (sensor specific)
5	*Event Data 1*
6	*Event Data 2*
7	*Event Data 3*

Table 162. CMM Status Sensor

ST	ED1	ERC	ED2	ED3	EC[a]	Event	SEL, SNMP Trap, and Health Event Output	(A)	(D)	SH
0xD9	01h	6Fh			0402	CMM Status Active: Assertion[b]	CMM Status Active	OK	-	yes
					0403	CMM Status Active: Deassertion (CMM Status Standby)[c]	CMM Status Active	-	OK	yes
	04h				0401	CMM Status Ready: Assertion	CMM Status Ready	OK	-	yes
					0400	CMM Status Ready: Deassertion (CMM Status Not Ready)	CMM Status Ready	-	Minor	yes
	0Eh				0404	CMM Status Ready Timeout: Assertion[d]	CMM Status Ready Timeout	Minor	-	yes
					0405[e]	CMM Status Ready Timeout: Deassertion (CMM Status Ready After Timing Out)	CMM Status Ready Timeout	-	OK	yes

a. Event Codes are in hexadecimal.
b. RSM transitions to the active state.
c. RSM transitions to the standby state.
d. Timeout expires before CMM becomes ready. Scripts triggered by this event will execute with some delay beyond the expiration of the timeout.
e. CMM becomes ready, but only after the timeout has expired.

Note: For information about setting the timeout mentioned in Table 162, see the `cmmreadytimeout` dataitem in "Alert Standard Format (ASF) Specification version 2.0".

D.30 HA Peer Lost Sensor

Table 163. HA Peer Lost Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
HA Peer Lost	D5h	70h	00h	00h		12E0	Redundancy regained or not active Shelf Manager	Redundancy regained or not active Shelf Manager	-	OK	yes
				01h		12E1	Connection with redundant peer lost due to CMM removal	Connection with redundant peer lost due to CMM removal	Major	-	yes
				02h		12E2	Connection with redundant peer lost due to CMM reboot or halt	Connection with redundant peer lost due to CMM reboot or halt	Major	-	yes

D.31 Power Restoration Failure

Table 164. Power Restoration Failure

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
Power Restoration Failure	D6h	70h	00h			1300	Power restore failure	Power restore failure; FRU HW address %1; FRU Device ID %2 where, %1 = IPMB Address from ED1 %2 = FRU ID from ED2	-	-	no

D.32 IPMC Reset Sensor

Table 165. IPMC Reset Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
IPMC Reset	EDh	03h	00h				Generates an event when the IPMC is reset		-	-	no

D.33 LMP Reset Sensor

Table 166. LMP Reset Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
LMP Reset	D4h	6Fh	01h				Generates an event when the LMP is reset		-	-	no

D.34 CFD Watchdog Sensor

Table 167. CFD Watchdog Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
CFD Watchdog	EEh	6Fh	00h				Event-only SDR type.		-	-	no

Note: Because it is an event-only sensor, the CFD Watchdog will not be listed in a `listtargets` report.

D.35 IPMC HA State Sensor

Table 168. IPMC HA State Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
IPMC HA State	D0h	6Fh	00h				Event is generated when the IPMC changes its redundant state. Event byte 2 is new state and event byte 3 is old state: 0x10 = active 0x03 = standby		-	-	no

D.36 IPMC Failover Sensor

Table 169. IPMC Failover Sensor

Sensor Type	STC	ERC	OF	ED2	ED3	EC	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
IPMC Failover	D1h	6Fh	00h				Event is generated when the IPMC begins failover, and another when failover processing is complete. Event byte 2 indicates failover state: 0 = failover start 1 = failover complete Event byte 3 indicates the failover reason for debug purposes: 1 = communication lost with active peer IPMC 2 = peer IPMC is not active 4 = Set Redundant Status command received 6 = both IPMCs are active		-	-	no

D.37 System Firmware Progress Sensor

Table 170. System Firmware Progress Sensor (sheet 1 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress						System Firmware Error (POST Error)	-	-	-	-
	System Firmware Error (POST Error)									
	0Fh	00h	00h		0250	- Unspecified (A)	System Firmware Error: Unspecified error occurred: Assertion	Major	-	Yes
						- Unspecified (D)	System Firmware Error: Unspecified error occurred: Deassertion	-	OK	Yes
			01h		0251	- No system memory physically installed (A)	System Firmware Error: No system memory installed: Assertion	Major	-	Yes
						- No system mem phys installed (D)	System Firmware Error: No system memory installed: Deassertion	-	OK	Yes
			02h		0252	- No usable sys mem - unrec failure (A)	System Firmware Error: No usable system memory found: Assertion	Major	-	Yes
						- No usable sys mem - unrec failure (D)	System Firmware Error: No usable system memory found: Deassertion	-	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 2 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	00h	03h		0253	- Unrecov HD/ ATAPI/IDE dev failure (A)	System Firmware Error: Unrecoverable hard disk/ ATAPI/IDE device: Assertion	Major	-	Yes
						- Unrecov HD/ ATAPI/IDE dev failure (D)	System Firmware Error: Unrecoverable hard disk/ ATAPI/IDE device: Deassertion	-	OK	Yes
			04h		0254	- Unrecoverable system-board failure (A)	System Firmware Error: Unrecoverable system-board failure: Assertion	Major	-	Yes
						- Unrecoverable system-board failure (D)	System Firmware Error: Unrecoverable system-board failure: Deassertion	-	OK	Yes
			05h		0255	- Unrecoverable diskette subsys failure (A)	System Firmware Error: Unrecoverable diskette subsystem failure: Assertion	Major	-	Yes
						- Unrecoverable diskette subsys failure (D)	System Firmware Error: Unrecoverable diskette subsystem failure: Deassertion	-	-	Yes
			06h		0256	- Unrecoverable HD controller failure (A)	System Firmware Error: Unrecoverable hard disk controller failure: Assertion	Major	-	Yes
						- Unrecoverable HD controller failure (D)	System Firmware Error: Unrecoverable hard disk controller failure: Deassertion	-	OK	Yes
			07h		0257	- Unrecoverable KB failure (A)	System Firmware Error: Unrecoverable PS/2 or USB keyboard failure: Assertion	Major	-	Yes
						- Unrecoverable KB failure (D)	System Firmware Error: Unrecoverable PS/2 or USB keyboard failure: Deassertion	-	OK	Yes
			08h		0258	- Removable boot media not found (A)	System Firmware Error: Removable boot media not found: Assertion	Major	-	Yes
						- Removable boot media not found (D)	System Firmware Error: Removable boot media not found: Deassertion	-	OK	Yes
			09h		0259	- Unrecoverable video controller failure (A)	System Firmware Error: Unrecoverable video controller failure: Assertion	Major	-	Yes
						- Unrecoverable video controller failure (D)	System Firmware Error: Unrecoverable video controller failure: Deassertion	-	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 3 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	00h	0Ah		025A	- No video device detected (A)	System Firmware Error: No video device detected: Assertion	Major	-	Yes
						- No video device detected (D)	System Firmware Error: No video device detected: Deassertion	-	OK	Yes
			0Bh		025B	- FW (BIOS) ROM corruption detected (A)	System Firmware Error: Firmware (BIOS) ROM corruption detected: Assertion	Major	-	Yes
						- FW (BIOS) ROM corruption detected (D)	System Firmware Error: Firmware (BIOS) ROM corruption detected: Deassertion	-	OK	Yes
			0Ch		025C	- CPU voltage mismatch (A)	System Firmware Error: CPU voltage mismatch: Assertion	Major	-	Yes
						- CPU voltage mismatch (D)	System Firmware Error: CPU voltage mismatch: Deassertion	-	OK	Yes
			0Dh		025D	- CPU speed matching failure (A)	System Firmware Error: CPU speed matching failure: Assertion	Major	-	Yes
						- CPU speed matching failure (D)	System Firmware Error: CPU speed matching failure: Deassertion	-	OK	Yes
			0E-98h		-	- Reserved	-	-	-	-
			99h	99h	0490	System Firmware Error: BIOS Checksum error	System Firmware Error: BIOS checksum error: [Assertion|Deassertion]	OK	OK	Yes
			9A-EFh		-	Reserved	-	-	-	-
			F0h	00h	027F	OK to boot	OK to boot: [Assertion|Deassertion]	OK	OK	Yes
			F1-FDh		-	Reserved	-	-	-	-
			FEh	00h	0280	System Firmware Error: Timer count read/write error	System Firmware Error: Timer count read/write error: [Assertion|Deassertion]	Critical	OK	Yes
				01h	0281	System Firmware Error: CMOS battery error	System Firmware Error: CMOS battery error: [Assertion|Deassertion]	Major	OK	Yes
				02h	0282	System Firmware Error: CMOS diagnosis error	System Firmware Error: CMOS diagnosis error: [Assertion|Deassertion]	Major	OK	Yes
				03h	0283	System Firmware Error: CMOS checksum error	System Firmware Error: CMOS checksum error: [Assertion|Deassertion]	Major	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 4 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	00h		04h	0284	System Firmware Error: CMOS memory size error	System Firmware Error: CMOS memory size error: [Assertion|Deassertion]	Major	OK	Yes
				05h	0285	System Firmware Error: RAM read/ write test error	System Firmware Error: RAM read/write test error: [Assertion|Deassertion]	Critical	OK	Yes
				06h	0286	System Firmware Error: CMOS date/ time error	System Firmware Error: CMOS date/time error: [Assertion|Deassertion]	Major	OK	Yes
				07h	0287	System Firmware Error: Clear CMOS jumper	System Firmware Error: Clear CMOS jumper: [Assertion|Deassertion]	OK	OK	Yes
				08h	0288	System Firmware Error: Clear password jumper	System Firmware Error: Clear password jumper: [Assertion|Deassertion]	OK	OK	Yes
				09h	0289	System Firmware Error: Manufacturing jumper	System Firmware Error: Manufacturing jumper: [Assertion|Deassertion]	OK	OK	Yes
				0Ah	028A	System Firmware Error: Microcontroller in update	System Firmware Error: Microcontroller in update: [Assertion|Deassertion]	Major	OK	Yes
				0Bh	028B	System Firmware Error: Microcontroller response failure	System Firmware Error: Microcontroller response failure: [Assertion|Deassertion]	Major	OK	Yes
				0Ch	028C	System Firmware Error: Event Log full	System Firmware Error: Event Log full: [Assertion|Deassertion]	OK	OK	Yes
				10h	028D	System Firmware Error: Configuration error on DIMM pair 0	System Firmware Error: Configuration error on DIMM pair 0: [Assertion|Deassertion]	OK	OK	Yes
				11h	028E	System Firmware Error: Configuration error on DIMM pair 1	System Firmware Error: Configuration error on DIMM pair 1: [Assertion|Deassertion]	OK	OK	Yes
				12h	028F	System Firmware Error: No system memory is physically installed or fails to access any DIMM's SPD data	System Firmware Error: No system memory is physically installed or fails to access any DIMM's SPD data: [Assertion|Deassertion]	OK	OK	Yes
			FFh		-	-	-	-	-	-

Table 170. System Firmware Progress Sensor (sheet 5 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	System Firmware Hang									
	0Fh	01h	00h		0460	- Unspecified (A)	System Firmware Hang: Unspecified error occurred: Assertion	Major	-	Yes
						- Unspecified (D)	System Firmware Hang: Unspecified error occurred: Deassertion	-	OK	Yes
			01h		0461	- Memory initialization (A)	System Firmware Hang: Memory initialization: Assertion	Major	-	Yes
						- Memory initialization (D)	System Firmware Hang: Memory initialization: Deassertion	-	OK	Yes
			02h		0462	- Hard-disk initialization (A)	System Firmware Hang: Hard disk initialization: Assertion	Major	-	Yes
						- Hard-disk initialization (D)	System Firmware Hang: Hard disk initialization: Deassertion	-	OK	Yes
			03h		0463	- Secondary processor(s) initialization (A)	System Firmware Hang: Secondary processor(s) initialization: Assertion	Major	-	Yes
						- Secondary processor(s) initialization (D)	System Firmware Hang: Secondary processor(s) initialization: Deassertion	-	OK	Yes
			04h		0464	- User authentication (A)	System Firmware Hang: User authentication: Assertion	Major	-	Yes
						- User authentication (D)	System Firmware Hang: User authentication: Deassertion	-	OK	Yes
			05h		0465	- User-initiated system setup (A)	System Firmware Hang: User-initiated system setup: Assertion	Major	-	Yes
						- User-initiated system setup (D)	System Firmware Hang: User-initiated system setup: Deassertion	-	OK	Yes
			06h		0466	- USB resource configuration (A)	System Firmware Hang: USB resource configuration: Assertion	Major	-	Yes
						- USB resource configuration (D)	System Firmware Hang: USB resource configuration: Deassertion	-	OK	Yes
			07h		0467	- PCI resource configuration (A)	System Firmware Hang: PCI resource configuration: Assertion	Major	-	Yes
						- PCI resource configuration (D)	System Firmware Hang: PCI resource configuration: Deassertion	-	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 6 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	01h	08h		0468	- Option ROM initialization (A)	System Firmware Hang: Option ROM initialization: Assertion	Major	-	Yes
						- Option ROM initialization (D)	System Firmware Hang: Option ROM initialization: Deassertion	-	OK	Yes
			09h		0469	- Video initialization (A)	System Firmware Hang: Video initialization: Assertion	Major	-	Yes
						- Video initialization (D)	System Firmware Hang: Video initialization: Deassertion	-	OK	Yes
			0Ah		046A	- Cache initialization (A)	System Firmware Hang: Cache initialization: Assertion	Major	-	Yes
						- Cache initialization (D)	System Firmware Hang: Cache initialization: Deassertion	-	OK	Yes
			0Bh		046B	- SM Bus initialization (A)	System Firmware Hang: SM Bus initialization: Assertion	Major	-	Yes
						- SM Bus initialization (D)	System Firmware Hang: SM Bus initialization: Deassertion	-	OK	Yes
			0Ch		046C	- KB controller init (A)	System Firmware Hang: Keyboard controller initialization: Assertion	Major	-	Yes
						- KB controller init (D)	System Firmware Hang: Keyboard controller initialization: Deassertion	-	OK	Yes
			0Dh		046D	- Embedded controller/ mgmt ctrller init (A)	System Firmware Hang: Embedded/Management controller initialization: Assertion	Major	-	Yes
						- Embedded controller/ mgmt ctrller init (D)	System Firmware Hang: Embedded/Management controller initialization: Deassertion	-	OK	Yes
			0Eh		046E	- Docking station attachment (A)	System Firmware Hang: Docking station attachment: Assertion	Major	-	Yes
						- Docking station attachment (D)	System Firmware Hang: Docking station attachment: Deassertion	-	OK	Yes
			0Fh		046F	- Enabling docking station (A)	System Firmware Hang: Enabling docking station: Assertion	Major	-	Yes
						- Enabling docking station (D)	System Firmware Hang: Enabling docking station: Deassertion	-	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 7 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	01h	10h		0470	- Docking station ejection (A)	System Firmware Hang: Docking station ejection: Assertion	Major	-	Yes
						- Docking station ejection (D)	System Firmware Hang: Docking station ejection: Deassertion	-	OK	Yes
			11h		0471	- Disabling docking station (A)	System Firmware Hang: Disabling docking station: Assertion	Major	-	Yes
						- Disabling docking station (D)	System Firmware Hang: Disabling docking station: Deassertion	-	OK	Yes
			12h		0472	- Calling operating system wake-up vector (A)	System Firmware Hang: Calling OS wake-up vector: Assertion	Major	-	Yes
						- Calling operating system wake-up vector (D)	System Firmware Hang: Calling OS wake-up vector: Deassertion	-	OK	Yes
			13h		0473	- Starting OS boot process (A)	System Firmware Hang: Starting OS boot process: Assertion	Major	-	Yes
						- Starting OS boot process (D)	System Firmware Hang: Starting OS boot process: Deassertion	-	OK	Yes
			14h		0474	- Baseboard/ motherboard init (A)	System Firmware Hang: Baseboard or motherboard initialization: Assertion	Major	-	Yes
						- Baseboard/ motherboard init (D)	System Firmware Hang: Baseboard or motherboard initialization: Deassertion	-	OK	Yes
			15h		N/A	- Reserved	-	-	-	-
			16h		0475	- Floppy init (A)	System Firmware Hang: Floppy initialization: Assertion	Major	-	Yes
						- Floppy init (D)	System Firmware Hang: Floppy initialization: Deassertion	-	OK	Yes
			17h		0476	- KB test (A)	System Firmware Hang: Keyboard test: Assertion	Major	-	Yes
						- KB test (D)	System Firmware Hang: Keyboard test: Deassertion	-	OK	Yes
			18h		0477	- Pointing device test (A)	System Firmware Hang: Pointing device test: Assertion	Major	-	Yes
						- Pointing device test (D)	System Firmware Hang: Pointing device test: Deassertion	-	OK	Yes

Table 170. System Firmware Progress Sensor (sheet 8 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	01h	19h		0478	- Primary processor init (A)	System Firmware Hang: Primary processor initialization: Assertion	Major	-	Yes
						- Primary processor init (D)	System Firmware Hang: Primary processor initialization: Deassertion	-	OK	Yes
			1Ah-FFh			- Reserved	-	-	-	-
	System Firmware Progress									
	0Fh	02h	00h		0260	- Unspecified (A)	System Firmware Progress: Unspecified error occurred: Assertion	OK	-	Yes
						- Unspecified (D)	System Firmware Progress: Unspecified error occurred: Deassertion	-	OK	
			01h		0261	- Memory initialization (A)	System Firmware Progress: Memory initialization: Assertion	OK	-	Yes
						- Memory initialization (D)	System Firmware Progress: Memory initialization: Deassertion	-	OK	
			02h		0262	- Hard-disk initialization (A)	System Firmware Progress: Hard disk initialization: Assertion	OK	-	Yes
						- Hard-disk initialization (D)	System Firmware Progress: Hard disk initialization: Deassertion	-	OK	
			03h		0263	- Secondary processor(s) initialization (A)	System Firmware Progress: Secondary processor(s) initialization: Assertion	OK	-	Yes
						- Secondary processor(s) initialization (D)	System Firmware Progress: Secondary processor(s) initialization: Deassertion	-	OK	
			04h		0264	- User authentication (A)	System Firmware Progress: User authentication: Assertion	OK	-	Yes
						- User authentication (D)	System Firmware Progress: User authentication: Deassertion	-	OK	
			05h		0265	- User-initiated system setup (A)	System Firmware Progress: User-initiated system setup: Assertion	OK	-	Yes
						- User-initiated system setup (D)	System Firmware Progress: User-initiated system setup: Deassertion	-	OK	

Table 170. System Firmware Progress Sensor (sheet 9 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	02h	06h		0266	- USB resource configuration (A)	System Firmware Progress: USB resource configuration: Assertion	OK	-	Yes
						- USB resource configuration (D)	System Firmware Progress: USB resource configuration: Deassertion	-	OK	
			07h		0267	- PCI resource configuration (A)	System Firmware Progress: PCI resource configuration: Assertion	OK	-	Yes
						- PCI resource configuration (D)	System Firmware Progress: PCI resource configuration: Deassertion	-	OK	
			08h		0268	- Option ROM initialization (A)	System Firmware Progress: Option ROM initialization: Assertion	OK	-	Yes
						- Option ROM initialization (D)	System Firmware Progress: Option ROM initialization: Deassertion	-	OK	
			09h		0269	- Video initialization (A)	System Firmware Progress: Video initialization: Assertion	OK	-	Yes
						- Video initialization (D)	System Firmware Progress: Video initialization: Deassertion	-	OK	
			0Ah		026A	- Cache initialization (A)	System Firmware Progress: Cache initialization: Assertion	OK	-	Yes
						- Cache initialization (D)	System Firmware Progress: Cache initialization: Deassertion	-	OK	
			0Bh		026B	- SM Bus initialization (A)	System Firmware Progress: SM Bus initialization: Assertion	OK	-	Yes
						- SM Bus initialization (D)	System Firmware Progress: SM Bus initialization: Deassertion	-	OK	
			0Ch		026C	- KB controller init (A)	System Firmware Progress: Keyboard controller initialization: Assertion	OK	-	Yes
						- KB controller init (D)	System Firmware Progress: Keyboard controller initialization: Deassertion	-	OK	
			0Dh		026D	- Embedded controller/ mgmt ctrller init (A)	System Firmware Progress: Embedded/ Management controller initialization: Assertion	OK	-	Yes
						- Embedded controller/ mgmt ctrller init (D)	System Firmware Progress: Embedded/ Management controller initialization: Deassertion	-	OK	

Table 170. System Firmware Progress Sensor (sheet 10 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	02h	0Eh		026E	- Docking station attachment (A)	System Firmware Progress: Docking station attachment: Assertion	OK	-	Yes
						- Docking station attachment (D)	System Firmware Progress: Docking station attachment: Deassertion	-	OK	
			0Fh		026F	- Enabling docking station (A)	System Firmware Progress: Enabling docking station: Assertion	OK	-	Yes
						- Enabling docking station (D)	System Firmware Progress: Enabling docking station: Deassertion	-	OK	
			10h		0270	- Docking station ejection (A)	System Firmware Progress: Docking station ejection: Assertion	OK	-	Yes
						- Docking station ejection (D)	System Firmware Progress: Docking station ejection: Deassertion	-	OK	
			11h		0271	- Disabling docking station (A)	System Firmware Progress: Disabling docking station: Assertion	OK	-	Yes
						- Disabling docking station (D)	System Firmware Progress: Disabling docking station: Deassertion	-	OK	
			12h		0272	- Calling operating system wake-up vector (A)	System Firmware Progress: Calling OS wake-up vector: Assertion	OK	-	Yes
						- Calling operating system wake-up vector (D)	System Firmware Progress: Calling OS wake-up vector: Deassertion	-	OK	
			13h		0273	- Stating OS boot process (A)	System Firmware Progress: Starting OS boot process: Assertion	OK	-	Yes
						- Stating OS boot process (D)	System Firmware Progress: Starting OS boot process: Deassertion	-	OK	
			14h		0274	- Baseboard/ motherboard init (A)	System Firmware Progress: Baseboard or motherboard initialization: Assertion	OK	-	Yes
						- Baseboard/ motherboard init (D)	System Firmware Progress: Baseboard or motherboard initialization: Deassertion	-	OK	
			15h		N/A	- Reserved	-	-	-	-

Table 170. System Firmware Progress Sensor (sheet 11 of 11)

Sensor Type	STC	OF	ED2[a]	ED3	EC[b]	Event	SEL, SNMP Trap, and Health Event Output	Severity (A)	Severity (D)	SH
System Firmware Progress	0Fh	02h	16h		0275	- Floppy init (A)	System Firmware Progress: Floppy initialization: Assertion	OK	-	Yes
						- Floppy init (D)	System Firmware Progress: Floppy initialization: Deassertion	-	OK	Yes
			17h		0276	- KB test (A)	System Firmware Progress: Keyboard test: Assertion	OK	-	Yes
						- KB test (D)	System Firmware Progress: Keyboard test: Deassertion	-	OK	Yes
			18h		0277	- Pointing device test (A)	System Firmware Progress: Pointing device test: Assertion	OK	-	Yes
						- Pointing device test (D)	System Firmware Progress: Pointing device test: Deassertion	-	OK	Yes
			19h		0278	- Primary processor init (A)	System Firmware Progress: Primary processor initialization: Assertion	OK	-	Yes
						- Primary processor init (D)	System Firmware Progress: Primary processor initialization: Deassertion	-	OK	Yes

a. ED2 provides an event extension code. (ED2 values of 15h and 1Ah–FFh are reserved values and do not appear in the table.)
b. Event Codes are in hexadecimal.

Appendix E Statistics

This appendix documents statistics that are implemented in the A6K-RSM-J shelf manager module firmware. Dash (–) means “not applicable”.

E.1 OS Statistics

Table 171. OS Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	OS	Load_Average_1	Average system load in the last minute	2nd order (AVG)	%	–	No
2		Load_Average_5	Average system load in the last 5 minutes	2nd order (AVG)	%	–	No
3		Load_Average_15	Average system load in the last 5 minutes	2nd order (AVG)	%	–	No
4		MemTotal	Total amount of memory	gauge	kBytes	–	No
5		MemFree	Free amount of memory	gauge	kBytes	–	No
6		DF_mtdblock<N>	File system free space (one statistic for each mounted JFFS file system)	gauge	%	–	No

E.2 Events Statistics

Table 172. Events Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	Event	EventsReceived	Number of received events	counter	–	–	Yes
2		CriticalEvents	Number of events recognized as critical severity	counter	–	–	Yes
3		MajorEvents	Number of events recognized as major severity	counter	–	–	Yes
4		MinorEvents	Number of events recognized as minor severity	counter	–	–	Yes
5		NormalEvents	Number of events recognized as normal severity	counter	–	–	Yes
6		UnknownEvents	Number of unrecognized events	counter	–	–	Yes
7		EventsDuplicated	Number of received duplicate events	counter	–	–	Yes
8		SelOverflows	Number of SEL overflows conditions	counter	–	–	Yes
9		SelResets	Number of SEL resets	counter	–	–	Yes
10		SelDrops	Number of dropped events due to SEL overflow	counter	–	–	Yes

E.3 Data Synchronization Statistics

Table 173. Data Synchronization Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	DataSync	BytesSent	Number of sent bytes	counter	Bytes	–	Yes
2		BytesReceived	Number of received bytes	counter	Bytes	–	Yes
3		BufferedDataSize	Size of currently buffered data	gauge	Bytes	–	Yes
4		FreeSmallBuffersLo	Number of small low priority free buffers	gauge	–	–	Yes
5		FreeSmallBuffersHi	Number of small high priority free buffers	gauge	–	–	Yes
6		FreeMediumBuffersLo	Number of medium low priority free buffers	gauge	–	–	Yes
7		FreeMediumBuffersHi	Number of medium high priority free buffers	gauge	–	–	Yes
8		FreeLargeBuffersLo	Number of large low priority free buffers	gauge	–	–	Yes
9		FreeLargeBuffersHi	Number of large high priority free buffers	gauge	–	–	Yes
10		SmallBufferPoolExhausted	Number of small buffer pool exhaust conditions	counter	–	–	Yes
11		MediumBufferPoolExhausted	Number of medium buffer pool exhaust conditions	counter	–	–	Yes
12		LargeBufferPoolExhausted	Number of large buffer pool exhaust conditions	counter	–	–	Yes
13		SuccessfulConnections	Number of successful connections	counter	–	–	Yes
14		TimeSinceLastConnection	Time since last successful connection	gauge	Seconds	–	Yes

E.4 IPMI Generic Statistics

Table 174. IPMI Generic Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thres-holds	Reset on Read
1	IpmiGeneric	RequestsDropped	Number of dropped requests	counter	–	–	Yes
2		RequestsEnqueued	Number of dropped requests	counter	–	–	Yes
3		RequestsDispatched	Number of all dispatched requests from IPMI clients	counter	–	–	Yes
4		RequestsDispatched_Shm	Number of dispatched requests from IPMI clients as SHM (source addr=20h)	counter	–	–	Yes
5		RequestsDispatched_Timed	Number of dispatched timed-out requests	counter	–	–	Yes
6		RequestsDispatched_Normal	Number of dispatched normal requests	counter	–	–	Yes
7		RequestsDispatched_System	Number of dispatched system requests	counter	–	–	Yes
8		ResponsesEnqueued	Number of enqueued responses	counter	–	–	Yes
9		ResponsesDispatched	Number of dispatched responses	counter	–	–	Yes
10		ResponsesDispatched_Local	Number of dispatched responses to local address	counter	–	–	Yes
11		ResponsesDispatched_Remote	Number of responses dispatched to remote address	counter	–	–	Yes
12		DispatchingQueue	Number of queue checks	counter	–	–	Yes
13		DispatchingQueue_NoAction	Number of queue checks without any action	counter	–	–	Yes
14		DispatchingQueue_Request	Number of dequeued requests	counter	–	–	Yes
15		DispatchingQueue_Response	Number of dequeued responses	counter	–	–	Yes
16		DispatchingQueue_Drop	Number of dropped requests due to aging	counter	–	–	Yes
17		RequestsReceived_NoHandler	Number of received requests without handler	counter	–	–	Yes
18		EventsReceived_NoSubscriber	Number of received events without subscriber	counter	–	–	Yes
19		ResponsesReceived_NoCallback	Number of received responses without callback	counter	–	–	Yes
20		RequestHandlerRegister	Number of request handler registrations	counter	–	–	Yes
21		EventSubscriberRegister	Number of event subscriber registrations	counter	–	–	Yes
22		RequestHandlerUnregister	Number of request handler deregistrations	counter	–	–	Yes
23		EventSubscriberUnregister	Number of event subscriber deregistrations	counter	–	–	Yes

Table 174. IPMI Generic Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thres-holds	Reset on Read
24		RequestCallbacksCancelled	Number of cancelled request callbacks	counter	–	–	Yes
25		RequestCallbacksCancel_NotFound	Number of request callbacks that were not cancelled because they were not found	counter	–	–	Yes
26		IpmbDrv_EventsReceived	Number of events received from IPMB driver	counter	–	–	Yes
27		IpmbDrv_RequestsReceived	Number of remote requests to addr 20h received from IPMB driver	counter	–	–	Yes
28		IpmbDrv_ResponsesReceived	Number of responses received from IPMB driver	counter	–	–	Yes
29		IpmbDrv_ResponseAcksReceived	Number of acknowledgements received from IPMB driver	counter	–	–	Yes

E.5 IPMI Message Pool Statistics

Table 175. IPMI Message Pool Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	IpmiMsgPool	MessagePoolBufferGet	Number of get buffer actions	counter	–	–	Yes
2		MessagePoolBufferRelease	Number of release buffer actions	counter	–	–	Yes

E.6 Cooling Statistics

Table 176. Cooling Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	Cooling	TemperatureEvents	Total number of received temperature events	counter	–	–	Yes
2		CriticalTemperatureEvents	Number of received critical temperature events	counter	–	–	Yes
3		MajorTemperatureEvents	Number of received major temperature events	counter	–	–	Yes
4		MinorTemperatureEvents	Number of received minor temperature events	counter	–	–	Yes
5		NormalTemperatureEvents	Number of received normal temperature events	counter	–	–	Yes

Table 176. Cooling Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
6		FruPowerReduce	Number of issued requests to reduce FRU power due to asserting major temperature condition	counter	–	–	Yes
7		FruPowerRestore	Number of issued requests to restore FRU power due to de-asserting major temperature condition	counter	–	–	Yes
8		FruDeactivate	Number of issued requests to deactivate FRU due to asserting critical temperature condition	counter	–	–	Yes

E.7 Local Sensor Repository Statistics

Table 177. Local Sensor Repository Statistics

No	Group Name	Statistic Name	Definition	Type	Unit	Supported Thresholds	Reset on Read
1	LSR	ShelfEventsAck	Number of acknowledged platform events for shelf sensors	counter	–	–	Yes
2		ShelfEventsNack	Number of unacknowledged platform events for shelf sensors	counter	–	–	Yes
3		LocalEventsAck	Number of acknowledged platform events for local sensors	counter	–	–	Yes
4		LocalEventsNack	Number of unacknowledged platform events for local sensors	counter	–	–	Yes
5		ShelfEventsSent	Number of sent platform events for shelf sensors	counter	–	–	Yes
6		LocalEventsSent	Number of sent platform events for local sensors	counter	–	–	Yes

Appendix F Legacy RPC Interface

The RSM can be administered by custom remote applications using remote procedure calls (RPC). RPCs provide all of the functionality of the CLI.

Remote Procedure Calls are useful for managing the RSM from:

- An administrator's computer using an in-house network
- Another blade in the same chassis as the RSM over the chassis backplane network
- An application running on the RSM itself

System Event Log (SEL) information is not available through the RPC interface.

F.1 Setting Up the RPC Interface

Before you can use RPC in a custom application, you must obtain the following C language RPC source code files:

- `rcliapi.h`
- `rcliapi_xdr.c`
- `rcliapi_clnt.c`
- `cli_client.h`
- `cli_client.c`

The first three files should be compiled and linked into your application program. These files implement the RPC calling subsystem for use in an application.

The file `cli_client.h` contains declarations and function prototypes necessary for interfacing with the RPC calling subsystem. Include the file with a `#include` directive in all the application files that make RPC calls.

The file `cli_client.c` contains a small sample program for calling the RSM through RPC that you can use for reference.

Note: These files can be downloaded as part of the CMM Software Development Kit. This kit is available from intel.driversdown.com.

F.2 Using the RPC Interface

The RPC interface may be used to manage the RSM whether the calling application is on a remote network, on a blade in the same chassis as the RSM, or even running on the RSM itself.

The following two functions are defined by the RPC subsystem for calling the RSM firmware:

- `GetAuthCapability()`
- `ChassisManagementApi()`

F.2.1 GetAuthCapability()

The following is the calling syntax for GetAuthCapability():

```
int GetAuthCapability(
      char* pszCMMHost,
      char* pszUserName,
      char* pszPassword
      );
```

Parameters
pszCMMHost: [in] IP Address or hostname of RSM
pszUserName: [in] A valid RSM user name
pszPassword: [in] Password associated with *pszUserName*

Return Value
>0 Authentication successful. The return value itself is the authentication code.

-1 Invalid username or password

E_RPC_INIT_FAIL RPC initialization failure.

E_RPC_COMM_FAIL RPC communication failure.

`GetAuthCapability()` is used to authenticate the calling application with the remote RSM. The remote RSM will not respond to RPC communications until the application has successfully authenticated. To authenticate, the application must pass the RSM's current IP address, login username, and login password to `GetAuthCapability()`.

The default username and password are `root` and `cmmrootpass`. When the authentication is successful, `GetAuthCapability()` returns an authentication code for use in all further RPC communications.

Note: Clients need to re-authenticate whenever the RSM is reset. Re-authentication is also necessary when the `ChassisManagementApi()` returns `E_ECMM_SVR_AUTH_CODE_FAIL`.

F.2.2 ChassisManagementApi()

The following is the calling syntax for `ChassisManagementApi()`:

```
int ChassisManagementApi(
      char*              pszCMMHost,
      int                nAuthCode,
      unsigned int       uCmdCode,
      unsigned char*     pszLocation,
      unsigned char*     pszTarget,
      unsigned char*     pszDataItem,
      unsigned char*     pszSetData,
      void **            ppvbuffer,
      unsigned int*      uReturnType
      );
```

Parameters

pszCMMHost [in] IP Address or DNS hostname of the RSM.

nAuthCode [in] Authentication code returned by `GetAuthCapability()`.

uCmdCode [in] The command to be executed (`CMD_GET` or `CMD_SET` as defined in `cli_client.h`).

pszLocation [in] The location that contains the dataitem that *uCmdCode* acts upon, such as `system`, `cmm`, or `blade1`.

pszTarget [in] The target that contains the attribute that *uCmdCode* acts upon, such as the sensor name as listed in the Sensor Data Record (SDR). When not applicable, use NA (such as when *pszDataItem* is an attribute of the *pszLocation* rather than *pszTarget.*)

pszDataItem [in] The attribute that *uCmdCode* acts upon, which is either an attribute of *pszLocation* or *pszTarget*.

pszSetData [in] The new value to set. When not applicable, use NA.

ppvbuffer [out] A pointer to the buffer containing the returned data.

uReturnType [out] The type of data that *ppvbuffer* points to. (See the `#define` directives in `cli_client.h`).

The value definitions of the return codes can be found in Table 178, “Error and Return Codes for the RPC Interface” on page 293.

Once the application has authenticated, it may proceed to get and set RSM parameters by calling `ChassisManagementApi()`. For each call to `ChassisManagementApi()`, the calling application must pass in the authentication code returned from `GetAuthCapability()`. The get and set commands available through `ChassisManagementApi()` are the same as those available through the CLI using `cmmget` and `cmmset`.

Note: SEL information is not available through the RPC interface.

Table 178. Error and Return Codes for the RPC Interface (sheet 1 of 7)

Code	Error Code String	Error Code Description
0	E_SUCCESS	Success
1	E_BPM_BLADE_NOT_PRESENT	Blade isn't in the chassis.
2	E_ECMM_SVR_COMMAND_UNSUPPORTED	ECMM_SVR: Unsupported Command Error.
3	E_CLI_MSG_SND	CLI Send Message Error.

Table 178. Error and Return Codes for the RPC Interface (sheet 2 of 7)

Code	Error Code String	Error Code Description
4	E_CLI_INVALID_TARGET	Not a valid -t parameter.
5	E_CLI_INVALID_LOCATION	Not a valid -l location.
6	E_CLI_INVALID_DATA_ITEM	Not a valid -d parameter.
7	E_CLI_INVALID_SET_DATA	Not a valid -v parameter.
8	E_CLI_INVALID_REQUEST	CLI Invalid Request Error.
9	E_CLI_MSG_RCV	CLI Receive Message Error.
10	E_CLI_NO_MORE_DATA	No data found to retrieve.
11	E_CLI_DATA_TYPE_UNSUPPORTED	CLI Data Type Unsupported.
12	E_ECMM_CLIENT_CONNECT_ERROR	ECMM_CLIENT: RPC Connect Error.
13	E_ECMM_SVR_AUTH_CODE_FAIL	Invalid auth code passed to RPC interface.
14	E_CLI_STANDBY_CMM	Operation cannot be performed on standby CMM.
15	E_WP_INITIALIZING	The CMM is Initializing and Not Ready. Wait a few seconds and try again.
16	E_BPM_NON_IPMI_BLADE	Blade does not support IPMI.
17	E_BPM_STANDBY_CMM	BPM operation cannot be performed on standby CMM.
18	E_BPM_NO_MORE_DATA	Couldn't delete a board from the drone mode list.
19	E_BPM_INVALID_SET_DATA	Not a valid -v parameter.
20	E_CLI_INVALID_BUFFER	Internal CMM Error.
21	E_CLI_INVALID_CMM_SLOT	Internal CMM Error.
22	E_CLI_NO_MSGQ_KEY	Internal CMM Error.
23	E_CLI_NO_MSGQ	Internal CMM Error.
24	E_CLI_NO_MSGQ_LOCK	Internal CMM Error.
25	E_CLI_NO_MSGQ_UNLOCK	Internal CMM Error.
26	E_CLI_FILE_OPEN_ERROR	Internal CMM Error.
27	E_CLI_CFG_WRITE_ERROR	CMM Config File Error.
28	E_IMB_NO_MSGQ	Internal CMM Error.
29	E_IMB_NO_MSGQ_KEY	Internal CMM Error.
30	E_IMB_SEND_TIMEOUT	Internal CMM Error.
31	E_IMB_DRIVER_FAILURE	Internal CMM Error.
32	E_IMB_REQ_TIMEOUT	A blade is not responding to IPMI requests.
33	E_IMB_RECEIVE_TIMEOUT	A blade is not responding to IPMI requests.
34	E_IMB_COMPCODE_ERROR	An IPMI request returned with a nonsuccessful completion code. User should try the command again.
35	E_IMB_INVALID_PACKET	Invalid IPMI response. Blade may be returning invalid data.
36	E_IMB_INVALID_REQUEST	Invalid IPMI response. Blade may be returning invalid data.
37	E_IMB_RESPONSE_DATA_OVERFLOW	Invalid IPMI response. Blade may be returning invalid data.
38	E_IMB_DATA_COPY_FAILED	Internal CMM Error.
39	E_IMB_INVALID_EVENT	Internal CMM Error.

Table 178. Error and Return Codes for the RPC Interface (sheet 3 of 7)

Code	Error Code String	Error Code Description
40	E_IMB_OPEN_DEVICE_FAILED	Internal CMM Error.
41	E_IMB_MMAP_FAILED	Internal CMM Error.
42	E_IMB_MUNMAP_FAILED	Internal CMM Error.
43	E_IMB_RESP_LEN_ERROR	Invalid IPMI response. Blade may be returning invalid data.
44	E_NEM_SNMPTRAP_ERROR	Error setting snmp trap parameters. Retry command.
45	E_NEM_SYSTEMHEALTH_ERROR	Internal CMM Error.
46	E_NEM_GETHEALTH_ERROR	Internal CMM Error.
47	E_NEM_SNMPENABLE_ERROR	Internal CMM Error.
48	E_NEM_SENSOR_HEALTH_ERROR	Internal CMM Error.
49	E_NEM_FILTER_SEL_ERROR	Internal CMM Error.
50	E_NEM_INITIALIZE_ERROR	Internal CMM Error.
51	E_NEM_SENSOR_EVENT	Internal CMM Error.
52	E_NEM_SENSOR_ERROR	Internal CMM Error.
53	E_NEM_SNMP_PROCESS_EVENT_ERROR	Internal CMM Error.
54	E_NEM_SNMP_DEST_ADDR_ERROR	SNMP Trap address that the user is setting is invalid.
55	E_NEM_SNMP_COMMUNITY_STRING_ERROR	SNMP Community that user is setting is invalid.
56	E_NEM_SNMP_TRAP_VERSION_ERROR	SNMP Trap version that the user is setting is invalid.
57	E_NEM_SNMP_TRAP_PORT_ERROR	SNMP Trap port that the user is setting is invalid.
58	E_NEM_SNMP_CFG_ERROR	Cannot read parameter. Configuration corrupted.
59	E_NEM_SEND_SNMP_TRAP_ERROR	Internal CMM Error.
60	E_SFS_INVALID_TRANSACTION	Internal CMM Error.
61	E_SFS_LOCK_SDR	Can't read SDRs. Blade may be busy, try again.
62	E_SFS_ENTITY_ID	Internal CMM Error.
63	E_SFS_DEVICE_LOCATOR_NULL	Internal CMM Error.
64	E_SFS_NO_MEMORY	Internal CMM Error.
65	E_SFS_UNSUPPORTED_DEVICE	Internal CMM Error.
66	E_SFS_RESPONSE_LENGTH	Internal CMM Error.
67	E_SFS_RESPONSE_DATA	Internal CMM Error.
68	E_SFS_POWER_SUPPLY_FRU	Internal CMM Error.
69	E_SFS_PATTERN_FOUND	Internal CMM Error.
70	E_SFS_SEMAPHORE_FAILED	Internal CMM Error.
71	E_SFS_CALLBACK_NOT_FOUND	Internal CMM Error.
72	E_SFS_END_OF_DATA	Internal CMM Error.
73	E_SFS_NO_SEL_ENTRY	Internal CMM Error.
74	E_SHEM_INTERNAL_ERROR	Internal CMM Error.
75	E_SHEM_INVALID_DATA_ITEM	Not a valid -d parameter.
76	E_SHEM_STANDBY_CMM	Cannot execute this command on the standby CMM.

Table 178. Error and Return Codes for the RPC Interface (sheet 4 of 7)

Code	Error Code String	Error Code Description
77	E_SNSR_STATUS_UNSUPPORTED	Internal CMM Error.
78	E_SNSR_UNSUPPORTED	Internal CMM Error.
79	E_SNSR_CATEGORY	Internal CMM Error.
80	E_SNSR_NO_MEMORY	Internal CMM Error.
81	E_SNSR_NOT_FOUND	Internal CMM Error.
82	E_SNSR_ACTION_UNSUPPORTED	Internal CMM Error.
83	E_SNSR_NON_FIRMWARE	Internal CMM Error.
84	E_SNSR_SHARE_CODE	Internal CMM Error.
85	E_SNSR_LOW_STORAGE	Internal CMM Error.
86	E_SNSR_EVENT_TYPE	Internal CMM Error.
87	E_SNSR_INVALID_REQUEST	Internal CMM Error.
88	E_SNSR_OS_ERROR	Internal CMM Error.
89	E_SNSR_PROCESSOR_NOT_PRESENT	Internal CMM Error.
90	E_SNSR_THRESHOLD_UNSUPPORTED	The sensor being queried doesn't support a particular threshold.
91	E_SNSR_CAPABILITY_UNSUPPORTED	Internal CMM Error.
92	E_SNSR_SCANNING_DISABLED	Internal CMM Error.
93	E_SNSR_MAX_RETRIES	Internal CMM Error.
94	E_SNSR_TRIGGER_TYPE	Internal CMM Error.
95	E_SNSR_STATE	Internal CMM Error.
96	E_SNSR_EVENT_DEREGISTER	Internal CMM Error.
97	E_SNSR_SEL_EVENT_FUNCTION	Internal CMM Error.
98	E_SNSR_BASE_INDEX	Internal CMM Error.
99	E_SNSR_PRESENCE_DETECTED	Internal CMM Error.
100	E_SNMP_CMD_UNSUPPORTED	Internal CMM Error.
101	E_SNMP_ERROR	Internal CMM Error.
102	E_SNSR_VALUE_OUT_OF_RANGE	Internal CMM Error.
103	E_SNSR_AUTH_ERROR	Internal CMM Error.
104	E_WP_INITIALIZE_LIBS	Internal CMM Error.
105	E_WP_CFG_READ_ERROR	CMM configuration file may be corrupted.
106	E_WP_CFG_WRITE_ERROR	CMM configuration file may be corrupted.
107	E_WP_THRESHOLD_UNSUPPORTED	The sensor being queried does not support a particular threshold.
108	E_WP_INVALID_TARGET	The sensor does not support a "current" value. This happens when querying a current value on a discrete sensor type.
109	E_WP_INVALID_LOCATION	Not a valid -l location.
110	E_WP_INVALID_DATA_ITEM	Not a valid -d parameter.
111	E_WP_INVALID_SET_DATA	Not a valid -v parameter.
112	E_WP_CMD_UNSUPPORTED	Not a supported command.
113	E_WP_STANDBY_CMM	Can't execute this command on the standby CMM.
114	E_WP_I2C_ERROR	Internal CMM Error.
115	E_FT_SEM_GET_FAILURE	Internal CMM Error.

Table 178. Error and Return Codes for the RPC Interface (sheet 5 of 7)

Code	Error Code String	Error Code Description
116	E_DRONE_NOT_FOUND	Internal CMM Error.
117	E_INTERNAL_ERROR	Internal CMM Error.
118	E_BPM_PWR_SUPPLY_NOT_PRESENT	Internal CMM Error.
119	E_NEM_INTERNAL_FAILURE	Internal CMM Error.
120	E_WP_CMM_RESET	CMM Reset.
121	E_UPDATE_INPROGRESS	Firmware update in progress.
122	E_CLI_INVALID_GET_DATA_ITEM	Not a valid getdataitem.
123	E_CLI_INVALID_SET_DATA_ITEM	Not a valid setdataitem.
124	E_SNSR_UPDATE_INPROGRESS	Sensor update in progress.
125	E_WP_SNSR_EVN_DESCRIPTION_NOT_FOUND	Sensor event description not found.
126	E_MSGQ_START	Message queue initializing. Retry operation.
127	E_PMS_ERROR	Process Management System error.
128	E_PMS_INVALID_RECOVERY_ACTION	Recovery action not allowed for this target.
129	E_CLI_MSG_RCV_TIMEOUT	Receive message timeout.
130	E_UPDATE_BADFRU	Chassis FRU cannot be read or is corrupted.
131	E_STANDBY_CMM_NOT_PRESENT	Standby CMM not present.
132	E_STANDBY_CMM_COMM_FAILURE	Failed to communicate with standby CMM.
133	E_FAILOVER_FAILED_BAD_SWITCH	Failover failed because of a bad switch.
134	E_FAILOVER_FAILED_BAD_NETWORK	Failover failed because of a bad network connection.
135	E_FAILOVER_FAILED_CRITICAL_EVENTS	Failover failed due to a critical event.
136	E_FAILOVER_FAILED_COMM_FAILED	Failover failed because of a communication failure.
137	E_FAILOVER_FAILED_UNHEALTHY	Failover failed because of an unhealthy event.
138	E_FAILOVER_FAILED_PRI1_NOT_SYNCED	Failover failed due to PRI1 not synching.
139	E_FAILOVER_FAILED_OLDER_FW_VERSION	Failover failed because the version of the other CMM's firmware is older.
140	E_FAILOVER_FAILED_STANDBY_STATE_UNKNOWN	Failover failed because the state of the standby CMM is unknown.
141	E_FAILOVER_FAILED	Failover failed.
142	E_CLI_SYNTAX_ERROR	CLI syntax error.
143	E_OS_ERROR	Operating system error.
144	E_CM_CONFIG_ERROR	Cooling Manager: Internal configuration error.
145	E_CM_NOT_NORMAL_LEVEL	Cooling Manager: Temperature level not normal.
146	E_CM_LC_NOT_ENABLED	Fantray does not support fantray control.
147	E_CM_NORMAL_TOO_HIGH	Cooling Manager: Cannot set the normallevel above the minorlevel.
148	E_CM_MINOR_TOO_HIGH	Cooling Manager: Cannot set the minorlevel above the maximumsetting.

Table 178. Error and Return Codes for the RPC Interface (sheet 6 of 7)

Code	Error Code String	Error Code Description
149	E_CM_NORMAL_TOO_LOW	Cooling Manager: Cannot set the normallevel below the minimumsetting.
150	E_CM_MINOR_TOO_LOW	Cooling Manager: Cannot set the minorlevel below the normallevel.
151	E_CM_COMM_FAILED	Cooling Manager: Communication with the fantray failed.
152	E_WP_FILE_NOT_FOUND	Action Scripts: File Not Found Error.
153	E_WP_SCRIPT_WAS_REMOVED	Action Scripts: Script Has Been Removed Error.
154	E_WP_SCRIPT_DIR_NOT_VALID	Action Scripts: Invalid Directory Error.
155	E_WP_DIR_NOT_ALLOWED	Action Scripts: Associating a Directory is Not Allowed Error.
156	E_WP_ZERO_SIZE	Action Scripts: Script is Zero (0) Size Error.
157	E_WP_NO_EXEC_PERMISSIONS	Action Scripts: No Owner Execute Permissions Error.
158	E_WP_ACTION_SCRIPTS_REMINDER	Action Scripts: Please, verify the script exists on the other CMM.
159	E_SUB_FRU_NOT_PRESENT	Sub-FRU Not Present.
160	E_NEM_GETUNHEALTHYFRUS_ERROR	Internal CMM Error.
161	E_NEM_GETNUMEVENTS_ERROR	Internal CMM Error.
162	E_NEM_CLEARHEALTH_ERROR	Internal CMM Error.
163	E_NEM_LOADHEALTH_ERROR	Internal CMM Error.
164	E_PROMOTE_SUCCESS	Standby CMM successfully promoted to active.
165	E_PROMOTE_FAILED_BAD_SWITCH	Promote cannot occur because the other CMM has a bad switch.
166	E_PROMOTE_FAILED_BAD_NETWORK	Promote cannot occur because the other CMM has lost network connectivity with its primary SNMP trap destination.
167	E_PROMOTE_FAILED_CRITICAL_EVENTS	Promote cannot occur because the standby CMM has critical health events.
168	E_PROMOTE_FAILED_COMM_FAILED	Promote cannot occur because the other CMM is not responding over its management bus.
169	E_PROMOTE_FAILED_PRI1_NOT_SYNCED	Promote cannot occur because the critical items have not been synched.
170	E_PROMOTE_FAILED_INCOMPATABLE_VERSIONS	Promote cannot occur because the standby has an older version of the firmware.
171	E_PROMOTE_FAILED_STANDBY_STATE_UNKNOWN	Promote cannot occur because the standby failover state discovery is not finished.
172	E_PROMOTE_FAILED_UNHEALTHY	Promote cannot occur because the other CMM has a bad hardware signal.
173	E_PROMOTE_FORCED_OCCURED	Standby CMM successfully promoted to active with forced option.
174	E_PROMOTE_FAILED_ACTIVE	Promote failed because it is executed on the active CMM.
175	E_PROMOTE_FORCED_OCCURED_COMM_FAILED	Promotion of standby CMM to active using forced option succeeded because the other CMM is not responding over its management bus.

Table 178. Error and Return Codes for the RPC Interface (sheet 7 of 7)

Code	Error Code String	Error Code Description
176	E_PROMOTE_FAILED	Promotion of standby CMM to active failed.
177	E_PROMOTE_FAILED_FAILOVER	Promotion of standby CMM to active failed because failover is in progress.
178	E_NW_ONLY_FRUUPDATE	Data updated only in the CDM and not in the backup files and the network stack.
179	E_NW_IP_UNDEFINED_IN_FRU	IP address value in CDM is undefined, set IP before setting this data.
180	E_NW_IP_RECORD_BASE_FORMAT	Only IP address value accepted since IP record in CDM is base format (version 00h).
181	E_BAD_BUFFER	Internal CMM Error.
(Unused)		
200	E_NOT_FOUND	Entity not found.
201	E_ILLEGAL_CMD_FOR_HA_STATE	Illegal command for HA state.
202	E_RPC_SVR_CONNECT_ERROR	Local RPC server connect rrror.
203	E_RPC_SVR_MISMATCH	Local RPC server version mismatch.
204	E_NO_PERM	Insufficient permissions.
205	E_THRESHOLD_UNSUPPORTED	Threshold unsupported.
206	E_NOT_SUBSCRIBED	Not subscribed.
207	E_ALREADY_SUBSCRIBED	Already subscribed.
208	E_CU_INVALID_DEST_ADDR_FORMAT	Upgrade Manager: Invalid destination address format.
209	E_CU_INVALID_FRU_TYPE	Upgrade Manager: Invalid FRU type.
210	E_CU_INVALID_DEST_HANDLE	Upgrade Manager: Invalid desination handle.
211	E_CU_INVALID_IMAGE_NAME	Upgrade Manager: Invalid image name.
212	E_CU_INVALID_IMAGE_INSTANCE	Upgrade Manager: Invalid image instance.
213	E_CU_INVALID_SOURCE	Upgrade Manager: Invalid source.
214	E_CU_INVALID_TYPE	Upgrade Manager: Invalid type.
215	E_CU_INVALID_PROTOCOL	Upgrade Manager: Invalid protocol.
216	E_CU_SRC_UNREACHABLE	Upgrade Manager: Source unreachable.
217	E_CU_SRC_CORRUPTED	Upgrade Manager: Source corrupted.
218	E_CU_DST_ACTIVE	Upgrade Manager: Destination active.
219	E_CU_INSUFFICIENT_SIZE	Upgrade Manager: Insufficient storage size.
220	E_CU_PROPERTY_NOT_SET	Upgrade Manager: Property not set.
221	E_CU_GET_PROPERTY_ERROR	Upgrade Manager: Property error.
222	E_CU_GET_PROPERTY_PARTIAL	Upgrade Manager: Invalid property.
223	E_CU_IMAGE_LOCKED	Upgrade Manager: Image already loaded.
224	E_CU_IMAGE_NOT_LOCKED	Upgrade Manager: Image not locked.
225	E_CU_IMAGE_VERIFICATION_ERROR	Upgrade Manager: Image verification error.
226	E_CU_RESTART_NOT_SUPPORTED	Upgrade Manager: Restart not supported.
227	E_CU_FUNCTION_NOT_SUPPORTED	Upgrade Manager: Function not supported.
228	E_CU_RESTART_INITIATED	Upgrade Manager: Restart Ininitiated.

F.2.3 ChassisManagementApi() threshold response format

Table 179, "Threshold Response Formats" lists the format of the ChassisManagementApi() queries that return data of type DATA_TYPE_ALL_THRESHOLDS.

Table 179. Threshold Response Formats

Dataitem	Return format	Example
thresholdsall	Data is returned in the THRESHOLDS_ALL structure as defined in cli_client.h. All structure fields are valid. If a particular threshold is not supported, the structure field contains an empty string. Each supported and valid field is a null-terminated string. **Syntax:** [*Value*] [*Units*] /n /0	5.400 Volts 5.200 Volts 5.100 Volts 4.600 Volts 4.800 Volts 4.900 Volts
uppernonrecoverable uppercritical uppernoncritical lowernonrecoverable lowercritical lowernoncritical	Data is returned in the THRESHOLDS_ALL structure defined in cli_client.h. Only the structure field corresponding to the dataitem requested is valid. If a particular threshold is not supported, the structure field contains an empty string. A valid field is a null-terminated string. **Syntax:** [*Value*] [*Units*] /n /0	5.160 Volts

F.2.4 ChassisManagementApi() string response format

Table 180, "String Response Formats" lists the format of ChassisManagementApi() queries that return data of type DATA_TYPE_STRING.

Table 180. String Response Formats (sheet 1 of 4)

Dataitem	Return Format	Example
current	Null-terminated string showing the current value of a sensor. **Syntax:** Value [Units] /0	23.000 Celsius
Ethernet	Null-terminated string showing the orientation of the eth0 Ethernet port: **Syntax:** [*front/back*] /0	front
healthevents	List of human-readable health events. Lines are separated by linefeeds with a null-terminator at the end. "(null)" or "" if there are no healthevents **Syntax:** [*Critical/Major/Minor*] Event: [*Health String*] /n /0	Minor Event: +3.3 V Upper non-critical going high asserted

Table 180. String Response Formats (sheet 2 of 4)

Dataitem	Return Format	Example
ListDataItems	List of available dataitems. Lines are separated by linefeeds and a null-terminator at the end. **Syntax:** [*Dataitem*] /n /0	presence listtargets listdataitems health healthevents sel snmpenable snmptrapcommunity snmptrapaddress1 snmptrapaddress2 snmptrapaddress3 snmptrapaddress4 snmptrapaddress5 redundancy powerstate
ListTargets	List of available targets. Targets represent the sensor data records (SDRs) for a particular component. Lines are separated by linefeeds with a null-terminator at the end. **Syntax:** [*Sensor Name*] /n /0	0:Brd Temp 0:+1.5 V 0:+2.5 V 0:+3.3 V 0:+5 V
ListLocations	List of available locations in the system. Except for the CMM locations are displayed as integers as follows: 1-14 = blade[1-14] 15 = Fantray1 16 = PEM1 17 = PEM2 CMM = CMM (only one CMM displayed)	CMM 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
location	Null-terminated string containing the user-specified physical location of the CMM, 16 characters maximum. **Syntax:** [*Location String*] /0	Server room 3
redundancy	Human-readable redundancy information containing the current CMM redundancy status. Lines are separated by linefeeds with a null-terminator at the end. **Syntax:** CMM 1: [*Present or Not Present*] ([*active or standby*]) [* *or no star*] /n CMM 2: [*Present or Not Present*] ([*active or standby*) [* *or no star*] /n * = The CMM you are logged into. /n /0	CMM 1: Present (active) * CMM 2: Not Present (standby) * = The CMM you are logged into.

Table 180. String Response Formats (sheet 3 of 4)

Dataitem	Return Format	Example
slotinfo	Human-readable slot information, containing a list of System slots, Peripheral slots, Busless slots, and Occupied slots. If there are no slots in a particular category, "None" is reported. Lines are separated by linefeeds with a null-terminator at the end. Each colon is followed by one tab (for Peripheral and Busless slots) or two tabs (for System and Occupied slots) and a space-delimited list of slot numbers. **Syntax:** System Slot(s): [*None or slot numbers*] /n Peripheral Slot(s): [*None or slot numbers*] /n Busless (Switch) Slot(s): [*None or slot numbers*] /n Occupied Slot(s): [*None or slot numbers*] / n /0	System Slot(s): None Peripheral Slot(s): 2 3 4 5 6 7 8 13 14 15 16 17 18 19 20 21 Busless (Switch) Slot(s): 2 19 20 21 Occupied Slot(s): 2 5 21
snmptrapaddress[*1..5*]	Null-terminated string containing a dotted-quad IP address **Syntax:** *aaa.bbb.ccc.ddd* /0	10.10.240.81
snmptrapcommunity	Null-terminated string containing the snmptrapcommunity name **Syntax:** *SNMP_Trap_Community_Name_String* /0	publiccmm
snmptrapport	Null-terminated string showing the SNMP trap port. **Syntax:** *port_number* /0	161
snmptrapversion	Null-terminated string showing the version of SNMP traps the CMM is currently set for. **Syntax:** [v1 *or* v3] /0	v3
version	Null-terminated string containing the version of the CMM firmware. **Syntax:** X.X.X.XXXX /0	5.1.0.117
AdminState	"1:Unlocked" or "2:Locked"	Used to set or query the administrative state of PMS as a whole, an individual monitored process. A target of "PmsGlobal" will set the state of the PMS as a whole. A target of PmsProc[#] will set the state of an individual process. "#" is the unique number of the process. AdminState is CMM-specific and is not synched between CMMs. It allows individual control of each CMM's adminstate and can be set on either the active or the standby CMM.
RecoveryAction	"1:No Action", "2:Process Restart", "3:Failover and Restart", or "4:Failover and Reboot"	Used to set or query the recovery action of a PMS monitored process. This is valid only for a target of PmsProc[#], where "#" is the unique number of the process.

Table 180. String Response Formats (sheet 4 of 4)

Dataitem	Return Format	Example
EscalationAction	"1:No Action", "2:Failover and Reboot"	Used to set or query the process restart escalation action. This is valid only for a target of "PmsProc[#] where "#" is the unique number of the process.
ProcessName	"<Process_Name> <Command_Line_Arguments>"	Used to query the process name and associated command line arguments for a monitored process. A target of "PmsProc[#] retrieves the name of an individual process where "#" is the unique number of the process.
OpState	"1:Enabled", "2:Disabled"	Used to query the operational state of a monitored process. An operational state of "2:Disabled" indicates that the process has failed and cannot be recovered. This is valid only for a target of PmsProc[#] where "#" is the unique number of the process.

F.2.5 ChassisManagementApi() integer response format

Table 181, "Integer Response Formats" lists the format of ChassisManagementApi() queries that return data of type DATA_TYPE_INT.

Table 181. Integer Response Formats

Dataitem	Return format	Example
health	Integer value corresponding to the health of the location queried: 0 = OK 1 = minor 2 = major 3 = critical	2
presence	Integer value corresponding to the absence or presence of the location queried: 0 = not present 1 = present If a blade is not present, ChassisManagementApi() returns E_BLADE_NOT_PRESENT.	1
snmpenable	Integer value indicating SNMP status: 0 = disabled 1 = enabled	0
powerstate	Integer value indicating the M-state of the location	4

F.2.6 FRU String Response Format

Querying an individual FRU field returns a null-terminated string where the last character of data in the string is the ASCII linefeed character. In other words, the last two bytes of the string contain the ASCII linefeed character and the ASCII null character.

Table 182. FRU Data Items String Response Format

Dataitem	Description of data returned in the string
all	All FRU information for the location.
boardall	All board area FRU information for the location.
boarddescription	Description field in the FRU board area for the location.
boardmanufacturer	Manufacturer field in the FRU board area for the location.
boardpartnumber	Part number field in the FRU board area for the location.
boardserialnumber	Serial number field in the FRU board area for the location.
boardmanufacturedatetime	Manufacture date and time field in the FRU board area for the location.
boardfrufileid	Lists the FRU file ID field in the board area for the location.
productall	product area FRU information for the location.
productdescription	description field in the FRU product area for the location.
productmanufacturer	Manufacturer field in the FRU product area for the location.
productmodel	Model field in the FRU product area for the location.
productpartnumber	Part number field in the FRU product area for the location.
productserialnumber	Serial number field in the FRU product area for the location.
productrevision	Revision field in the FRU product area for the location.
productassettag	Lists the asset tag field in the FRU product area for the location
chassisall	All chassis area FRU information for the location.
chassispartnumber	Part number field in the FRU chassis area for the location.
chassisserialnumber	Serial number field in the FRU chassis area for the location.
chassislocation	Location field in the FRU chassis area for the location.
chassistype	Type field in the FRU chassis area for the location.
listdataitems	List of all of the FRU dataitems that can be queried for the FRU target.

F.3 RPC Sample Code

Sample code for interfacing with the RSM through RPC is available in the file `cli_client.c`. The compiled output of the sample code is a command-line executable for use on the Linux operating system or an object file (`*.o` file) for use on the VxWorks operating system. To select a given target, uncomment the appropriate `#define` directive in the source code.

The sample code first authenticates with the RSM by calling `GetAuthCapability()`. When authentication is successful, the user's command-line arguments (for Linux) or calling parameters (for VxWorks) are passed to the RSM by calling `ChassisManagementApi()`. The return code is then checked and the result is printed to the console.

F.4 RPC Usage Examples

Table 183 presents examples of using RPC calls to get and set fields on the RSM. Data returned by RPC calls are held in the *ppvbuffer* and *uReturnType* parameters associated with the function `ChassisManagementApi()`.

Table 183. RPC Usage Examples (sheet 1 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Get the chassis temperature	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: Chassis *pszTarget*: TempSensorName *pszDataItem*: current	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A null-terminated string of the format: Value [Units]
Get the fan tray presence	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: fantray1..3 *pszTarget*: NA *pszDataItem*: presence	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value indicating presence 1 = Present 0 = Not Present
Get the CPU temperature of blade 5	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade5 *pszTarget*: CPUTempSensorName *pszDataItem*: current	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A null-terminated string of the format: Value [Units]
Determine if a certain blade is present	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade[1-n] *pszDataItem*: presence	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Present The call to ChassisManagementApi() returns E_BLADE_NOT_PRESENT if the selected blade is not present.
Get all thresholds for the +3.3 V sensor on blade 2	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade2 *pszTarget*: 3.3vSensorName *pszDataItem*: ThresholdsAll	*uReturnType*: DATA_TYPE_ALL_THRESHOLDS *ppvbuffer*: A THRESHOLDS_ALL structure as defined in cli_client.h
Get the overall system health	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: system *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get a list of blades with problems	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: system *pszDataItem*: unhealthylocations	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: List of all blades with problems
Get the temp1 sensor's health on blade 5	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade5 *pszTarget*: Temp1SensorName *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get the CMM's overall health	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: CMM *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical

Table 183. RPC Usage Examples (sheet 2 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Get a blade's overall health	*pszCMMHost*: localhost *uCmdCode* CMD_GET *pszLocation*: blade[1..n] *pszDataItem*: health	*uReturnType*: DATA_TYPE_INT *ppvbuffer*: Integer value denoting health state 0 = OK 1 = Minor 2 = Major 3 = Critical
Get the version of software on the CMM	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: CMM *pszDataItem*: version	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A human-readable null-terminated version string.
Power off one of the blades	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: poweroff	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Power on one of the blades	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: poweron	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Reset a blade	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: blade[1-19] *pszDataItem*: powerstate *pszSetData*: reset	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Determine what sensors are on blade 3	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade3 *pszDataItem*: ListTargets	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A list of sensor names as defined in the SDR.
Determine what may be queried or set on a blade	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade3 *pszDataItem*: ListDataItems	*uReturnType*: DATA_TYPE_STRING *ppvbuffer:* A list of commands to be used as data items.
Determine what may be queried on the blade4 +3.3 V sensor	*pszCMMHost*: localhost *uCmdCode*: CMD_GET *pszLocation*: blade4 *pszTarget*: +3.3SensorName *pszDataItem*: ListDataItems	*uReturnType*: DATA_TYPE_STRING *ppvbuffer*: A list of commands to be used as data items.
Enable the SNMP Traps	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPEnable *pszSetData*: enable	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Set the SNMP Target	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPTrapAddress[1-5] *pszSetData*: 134.134.100.34	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.

Table 183. RPC Usage Examples (sheet 3 of 3)

Example	ChassisManagementApi() [in] Parameters	ChassisManagementApi() [out] Parameters
Set the SNMP Community	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: chassis *pszDataItem*: SNMPCommunity *pszSetData*: public	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Set the Telco Alarm on	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: CMM *pszDataItem*: TelcoAlarm *pszSetData*: 1	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.
Light Major LED on the CMM	*pszCMMHost*: localhost *uCmdCode*: CMD_SET *pszLocation*: CMM *pszDataItem*: MajorLED *pszSetData*: 1	*uReturnType*: not used *ppvbuffer*: not used The return code from ChassisManagementApi() indicates success or failure.

Appendix G Reference Information

This appendix provides links to data sheets, standards, and specifications for the technology designed into the A6K-RSM-J shelf manager module.

G.1 AdvancedTCA* Product Information

Information and software updates can be found for AdvancedTCA products from Radisys at:

http://www.radisys.com

G.2 AdvancedTCA Specifications

Current AdvancedTCA Specifications can be purchased from PICMG for a nominal fee. Short form specifications in Adobe Acrobat format (PDF) are also available on the PICMG website at:

http://www.picmg.org/pdf/PICMG_3_0_Shortform.pdf

G.3 IPMI

Current specifications for the Intelligent Platform Management Interface (IPMI) can be found at:

http://developer.intel.com/design/servers/ipmi/spec.htm

Appendix H ShMgr Version Feature Differences

This appendix describes the features and functionality for ShMgr software version 8.x that differ from version 7.1.x. The A6K-RSM-J shelf manager module uses ShMgr software version 8.x.

H.1 LISM

H.1.1 ShMgr software 7.1.x is designed to be a Location Independent Shelf Manager (LISM)

H.1.2 For version 8.x, the "software IPMC process" and associated functionality are decoupled from the LISM

H.2 Porting to version 8.1.X includes porting ShMgr software to a different platform

H.2.1 Wind River 3.0

Wind River 3.0 replaces the open source version of Linux.

H.2.2 New LMP processor

The LMP for version 8.x is the Freescale P2020 32-bit QorIQ processor:

H.2.3 New IPMC

The version 8.x IPMC is powered by the Renesas H8/2472.

H.2.4 U-Boot firmware bootstrapping

A U-Boot firmware image replaces RedBoot for bootstrapping the embedded environment once power is applied to the chassis.

H.3 Shelf management functionality is divided into two distinct components

Version 8.x divides shelf management operation into these separate components:

H.3.1 Low-level code running on the Renesas H8S/2472 microcontroller (ShMC)

H.3.2 High-level code running on a Local Management Processor (LMP)

The shelf management controller and LMP components communicate with each other over the system interface. Any hardware which provides these components is capable of hosting the shelf management solution.

H.4 Cannot upgrade from ShMgr versions 5.2.x, 6.1.x, and 7.1.x

ShMgr software version 8.x does not provide upgrade support for earlier ShMgr software versions 5.2.x, 6.1.x, and 7.1.x.

H.5 FRU power management

Power budget prioritization logic puts the subFRUs at the top of the power budgeting queue, getting power assigned first before powering main FRUs of other IPMCs.

FRUs which depend on a powered subFRU by the time their operating systems are initializing, such as hard disk drives, PCI express, etc., will boot properly with all dependencies satisfied.

H.6 Performance improvements

H.6.1 Event management

Event management is improved through these modifications:

- Enhanced the ability of the LISM to process more events and IPMI requests
- Prevented the overloading of incoming events and IPMI requests while the LISM is booting up and not ready to receive or process events or requests
- Increased the queue size for incoming events
- Added a second thread for quicker processing of events and requests
- Fewer SDR reloads from the same IPMC

H.6.2 SDR management

SDR loading is streamlined with additional logic that provides these benefits:

- Quicker SDR load time
- Fewer SDR load retries
- Fewer SDR reloads from the same IPMC